The Effects of Cooperative Learning on Content Comprehension and Logical Reasoning in a Second-Semester University Computer Science Course

Approved by
Dissertation Committee:

Lowell J. Bethel

Nell Dale

[illegible]

Sharon E. Nichols

James P. Barufaldi

The Effects of Cooperative Learning on Content Comprehension and Logical Reasoning in a Second-Semester University Computer Science Course

by

Roger Louis Priebe, B.S., M.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

August 1997

To my family

Acknowledgments

I would like to thank my advisor Dr. Lowell Bethel, along with the other members of my committee, Dr. Nell Dale, Dr. John Huntsberger, Dr. Jim Barufaldi, and Dr. Sherry Nichols for their help, constructive criticism, and patience during this study. In addition, I am especially grateful to the members of the computer science education group and in particular Suzy Gallagher, Dr. Vicki Almstrum, Dr. Henry Walker, and Dr. Barbara Boucher-Owens.

I would like to thank my friends, classmates, and family for their support and encouragement. A special thanks to Helen Kluna for helping me negotiate the paperwork maze and for her kind heart. Finally, I would like to thank Beth Macom for her constant support and perspicacity.

The Effects of Cooperative Learning on Content Comprehension and Logical Reasoning in a Second-Semester University Computer Science Course

Publication No._______________

Roger Louis Priebe, Ph.D.

The University of Texas at Austin, 1997

Supervisor: Lowell J. Bethel

Attrition rates in the computer science major are quite high. Many students who struggle through the first few courses ultimately drop out of the major when the coursework becomes too complex, mostly because of the increased amount of logic and abstraction that the coursework requires.

This study compared content comprehension, logical reasoning ability, and attendance in two groups of second-semester university computer science students. In a quasi-experimental, pretest/posttest, control-group design, the control group (n=25) received instruction in a traditional lecture/discussion learning environment three days a week for nine weeks. The treatment group (n=24) met in a cooperative learning environment for the same number of hours as the control group. Each group was given the pretest and posttest for the Burton Comprehension Instrument (BCI) and a pretest and posttest for the Propositional Logic Test (PLT) to measure levels of content comprehension and logical reasoning ability. A head count was taken daily to determine if the cooperative learning environment might promote better attendance.

The null hypotheses investigated in this study were: (1) There will be no difference between the cooperative learning and control groups in concept comprehension; (2) There will be no difference between the cooperative learning and control groups in the improvement of logical thinking skills; and (3) There will be no difference between the cooperative learning and control groups in attendance. The collected data were analyzed by the use of Analysis of Covariance (ANCOVA), Multivariate Analysis of Variance (MANOVA) with Repeated Measures, and one-way Analysis of Variance (ANOVA).

The results of the analysis revealed no difference between the cooperative learning and lecture groups in the areas of content comprehension or logical reasoning ability. However, the cooperative learning group did have significantly better attendance ($p<.03$).

Further research is recommended in the use of cooperative learning in university-level computer science courses. Of special interest are research on the use of cooperative learning techniques in large lecture-based courses and research on the effect of cooperative learning on gender equity in computer science.

Table of Contents

List of Tables	xiii
List of Figures	xv
Chapter 1. The Problem	1
Introduction	1
Background	2
Statement of the Problem	3
Purpose of the Study	3
Research Questions	4
Rationale and Theoretical Base	4
The Lecture Method	6
The Cooperative Learning Method	6
Research Hypotheses	8
Achievement	8
Logical Reasoning	9
Attendance	12
Importance of the Study	13
Achievement	14
Logical Reasoning Ability	14
Definition of Terms	15
Summary	16
Chapter 2. Review of the Literature	18
Theoretical Foundations	18

Jean Piaget 19

Lev Vygotsky 21

Pedagogy and Introductory Computer Science 23

The Role of Logic in the Curriculum 24

Other Computing Curriculum 1991 Recommendations 26

Learning Paradigms 27

The Lecture Method 27

The Cooperative Learning Method 28

Research on the Cooperative Learning Paradigm 33

Positive Aspects of Cooperative Learning 33

Collaboration Within Cooperative Learning 37

Cooperative Learning in University-Level Computer Science 38

Undergraduate Student Attendance 42

Summary 43

Chapter 3. Methodology 45

Research Hypotheses 46

Population Sample 46

Control Group 48

Treatment Group 48

Final Sample After Attrition 49

Dependent Variables 50

Comprehension 50

Logical Reasoning 50

Attendance 51

Independent Variable 51

Learning Environment 51

Methodology	51
The Lecture/Discussion Environment	53
The Cooperative Learning Environment	54
Cooperative Environment Validation	59
Instrumentation	60
Burton Comprehension Instrument	60
Propositional Logic Test	62
Other Data Gathering	64
Questionnaire	64
Attendance	64
Group Evaluations	64
Course Evaluations	64
Data Analysis	65
Comprehension	65
Logical Reasoning	65
Attendance	65
Limitations of the Investigation	66
Summary	67
Chapter 4. Results	68
Overview of Data Analysis	68
Population Sample	68
Results of Data Analysis	69
Concept Comprehension	69
Logical Reasoning	71
Attendance	75
Summary of Findings	76

Chapter 5. Discussion 78

Summary 78

Results and Discussion 79

Content Comprehension 79

Logical Reasoning 81

Attendance 84

Qualitative Observations 85

Conclusions 87

Future Research 88

Appendices 92

Appendix A Consent Form 92

Appendix B Questionnaire 94

Appendix C Burton Comprehension Instrument (BCI) Pretest 96

Appendix D CS 315 Final Examination with Embedded BCI Posttest 102

Appendix E Propositional Learning Test with Instructions 117

Appendix F Alternate Propositional Learning Test (A-PLT) 117

Appendix G Group Evaluation Form 130

Appendix H External Cooperative Learning Validation 132

Appendix I Attendance Statistics 138

Bibliography 140

Vita 151

List of Tables

Table 2.1	Reported Pretest and Posttest Means from Sabin and Sabin (1994)	40
Table 4.1	Means and Standard Deviations for the BCI Pretest and Posttest	70
Table 4.2	ANCOVA Result Comparing the Cooperative Learning and Control Groups on the Burton Comprehension Instrument (BCI)	71
Table 4.3	Means and Standard Deviations for the PLT Pretest and Posttest	72
Table 4.4	MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the Propositional Logic Test (PLT)	73
Table 4.5	Means for the Pretest and Posttest PLT Subtests	73
Table 4.6	MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Conjunction Subtest	74
Table 4.7	MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Disjunction Subtest	74
Table 4.8	MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Implication Subtest	74
Table 4.9	MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Biconditional Subtest	74
Table 4.10	Means and Standard Deviations on the Attendance Between Classes (N=23 lectures)	76
Table 4.11	ANOVA Result on the Attendance Between Classes (N=23 lectures)	76
Table 4.12	Summary of Hypothesis Testing	77

Table 5.1	Means and Standard Deviations for the Midterm Examinations	80
Table 5.2	Pretest and Posttest Means on the Propositional Logic Test (PLT)	82

List of Figures

Figure 1.1 Graphical Representation of Cooperative Learning and the Resulting Increase in Logical Reasoning Ability 11

Chapter 1. The Problem

Introduction

Life is a cooperative effort. Each day we all depend upon others for food, companionship, and most of our basic needs. Even many activities that seem to be competitive in nature—such as games or sporting events—ultimately become a form of cooperation as we agree to play by the same rules and maintain a common discourse. Gaining knowledge from our surroundings is a never-ending process that contributes to who we are and how we contribute to our society. Through this form of cooperation we define ourselves and are defined by others.

Although many of the world's activities are conducted cooperatively, the pedagogy used in higher education remains quite competitive or individualistic (Johnson, Johnson, & Smith, 1991). Teachers generally teach as they were taught (typically using a traditional lecture format), which makes change in academia a slow process. Change does take place, however, and strategies for cooperative learning are being used successfully at the college level as an alternative to the current lecture-based learning paradigm (Purdom & Kromrey, 1995).

Background

Computer science is a highly complex and abstract subject matter. The comprehension and retention of many key concepts in computer science rely on students' ability to engage in formal reasoning. This prerequisite can cause problems for beginning computer science students, however. Toothacker (1983) found that only one-third of entering university students were capable of such formal reasoning.

Almstrum (1994) found that novice computer science students experienced more difficulty with concepts involving mathematical logic than with more general computer science concepts. Kim (1995) concluded that propositional reasoning ability—a facet of formal-operational thinking—was related to course achievement in a logic class for computer science majors. Surprisingly, she also found that a course in logic that was specifically designed to teach the logical concepts measured in the study did not improve students' performance. One of her recommendations was to study new ways to teach introductory computer science courses, particularly the logic courses.

Studies have concluded that three of the most important activities that can increase students' thinking skills are student discussion, explicit emphasis on problem solving using varied methods and examples, and verbalization of methods and strategies to encourage the development of metacognition. These activities are all integral to a cooperative learning environment. Studies have shown that cooperative learning promotes a greater use of higher-level reasoning strategies and critical thinking than do competitive or individualistic learning (Gabbert, Johnson, & Johnson, 1986; McKeachie, 1988; Skon, Johnson, & Johnson, 1981; Slavin, 1985).

Statement of the Problem

What are the effects of cooperative learning on content comprehension and logical reasoning in a second-semester university computer science course?

Purpose of the Study

Attrition rates in the computer science major are quite high. Many students who struggle through the first few courses ultimately drop out of the major when the coursework becomes too complex. Much of this complexity is due to the increased amount of logic and abstraction required for the courses.

Because of the structure of higher education, computer science education typically begins with students who are 17 or 18 years old. As suggested by the results of recent studies (Almstrum, 1994; Kim, 1995; Toothacker, 1983), not only are many students not yet capable of the formal reasoning required for success in computer science, but content courses specifically designed to remediate this problem are not helpful. This poses a pedagogical problem for computer science education. The purpose of this study is to investigate whether cooperative learning is a useful tool for remediating the deficiencies demonstrated by many beginning computer science students.

Research Questions

1. Will students in a cooperative learning environment comprehend computer science content better than students in a traditional lecture course?

2. Will cooperative learning create an environment that helps students move through Piaget's cognitive stages and thus improve their logical thinking skills?

Of secondary interest:

3. Will a cooperative learning environment produce better attendance than a traditional lecture course?

Rationale and Theoretical Base

Interactions among students can be classified in one of three ways: competitive, individualistic, or cooperative (Johnson & Johnson, 1994). A competitive situation is one in which a student's success depends upon the failure of another student, such as in the case where a teacher grades "on the curve" and only a limited number of high grades are "available." In an individualistic learning situation, no direct competition for grades exists, but the students work by themselves with little or no collaboration. Finally, a cooperative learning environment exists where members of a group work toward a common goal, are committed to maximizing the learning potential of each member of the group, and share rewards.

College instruction has been criticized for not preparing students to engage in the world as it actually exisits—merely rewarding students for short-term and shallow retention of information. This criticism is compounded by the incongruity between the ideal world of educational theory and its actual practice (Lazar, 1995). Much of this problem lies in educators' inability to move away from traditional forms of teaching and learning, with their problematic lack of attention paid to classroom dynamics and inter-student interaction (Johnson, Johnson, & Smith, 1991).

In the traditional educational paradigm, knowledge is treated as a "substance" that is transferred from teachers to students. The student is seen as a passive vessel needing to be filled with the knowledge held by the instructor (Johnson, et al., 1991). No student-student interaction is required for the transmission of knowledge. This paradigm is being challenged by another that posits the idea that students actively construct their own knowledge and that this knowledge is discovered, transformed, and extended by students in interactions with other students, faculty members, and their environment (Johnson, et al., 1991; Roth, 1990).

The Lecture Method

The lecture method is currently the most commonly used paradigm in college teaching (Johnson, et al., 1991). Lectures are easily adaptable to different audiences and the amount of time a lecture consumes is easily controlled. The main problem with a lecture pedagogy is that the learner is made to be a passive receiver of knowledge. Due to enforced passivity, student attention often wavers, causing lecture material to be missed or remembered incorrectly (Penner, 1984). In addition to a lack of attention, passive learning results in a lower level of learning as categorized by Bloom's Taxonomy (Bloom, 1956). McKeachie and Kulik (1975) found that while lecturing was effective for the transmission of factual knowledge, discussion was a superior means of promoting higher-level reasoning and problem solving.

The Cooperative Learning Method

A cooperative learning environment is characterized by small groups of students working together toward group and individual goals (Slavin, 1990). Within these groups, students provide academic and social support while engaging in an active learning environment. Student-student and student-instructor interactions provide the necessary communication to promote learning and create a community of learners. Researchers have found many advantages to cooperative learning over the traditional competitive or individualistic environments: increased achievement and critical thinking competency, better attitudes toward the subject area, enhanced interpersonal relationships, higher self-esteem, lower attrition rates, and better cross-cultural relations (Gabbert, et al., 1986; Grisham & Molinelli, 1995; Johnson, et al., 1991, Johnson & Johnson, 1994; Skon, et al., 1981; Slavin, 1990; Treisman, 1985; Wales & Sager, 1978).

Students are likely to seek more information from their classmates in a cooperative environment than in a competitive one, a factor that promotes a more efficient exchange of information. This cooperation often takes the form of *peer teaching*—a situation in which a more knowledgeable peer shares information and ideas with others. The process of peer teaching creates a bond of trust within the group and further helps motivate students to achieve. This trust also allows the students to challenge each others' conclusions and contribute ideas without fear of retribution, creating a fertile problem-solving environment (Johnson & Johnson, 1994).

The Joint Task Force of the Association of Computer Machinery/Institute of Electrical and Electronics Engineers-Computer Science (ACM/IEEE-CS) set forth a number of goals in its *Computing Curricula 1991* (Tucker, 1991). Among these goals are the ability to define a problem clearly, determine its tractability, choose an appropriate solution strategy, and test the solution strategy. This process involves collaboration with other professionals and the ability to work in a team environment throughout the entire problem-solving process. These are precisely the skills that a cooperative learning environment fosters, as described by Johnson, et al. (1991):

> Cooperative learning is indicated whenever the learning goals are highly important, mastery and retention is important, the task is complex or conceptual, problem solving is desired, divergent thinking or creativity is desired, quality of performance is expected, and higher-level reasoning strategies and critical thinking are needed. (p. 2:13)

Research Hypotheses

Achievement

- Will students in a cooperative learning environment comprehend computer science content better than students in a traditional lecture course?

In the last 90 years, nearly 400 studies have been conducted to determine if cooperative learning promotes higher productivity and achievement. A meta-analysis of these studies shows that in general students score roughly 2/3 of a deviation higher in a cooperative learning environment than in either a competitive or individualistic environment (Johnson & Johnson, 1989). Therefore, it is expected that the students in the cooperative learning environment will comprehend the material better and show greater achievement than the students in the traditional lecture environment. This research question will be examined with the null hypothesis $H_0 1$.

$H_0$1: There will be no difference between the cooperative learning and control groups in concept comprehension.

Logical Reasoning

- Will cooperative learning foster higher-order thinking skills necessary to move students through cognitive phases and hence improve their logical thinking skills?

Research has shown that successful group learning can foster beneficial consequences other than greater achievement. Group learning situations promote more student exchanges that enhance reasoning and higher-order thinking (Bossert, 1988-1989). Students also benefit when they share ideas, accommodate others' perspectives, and give and receive help. These benefits are more likely to occur when the task entails problem solving and has more than one potentially correct answer (Blumenfeld, Marx, Soloway, & Krajcik, 1996). This research corroborates the work of McKeachie (1988) who concluded that three of the most important activities that can increase students' thinking skills are student discussion, explicit emphasis on problem solving using varied methods and examples, and verbalization of methods and strategies to encourage the development of metacognition.

The potential benefits of cooperative learning on logical reasoning have a psychological foundation. Although not in agreement on the mechanisms of the process of intellectual growth, Jean Piaget and Lev Vygotsky both viewed intellectual improvements as qualitative and not quantitative, seeing quality of thought and the ability to gain new knowledge as more important than sheer amount of knowledge. This quality is expressed in Piaget's stages of intellectual growth and the transitions between these stages. Importantly, the final formal operational stage is defined primarily by the ability to perform logical reasoning and other abstractions including the ability to conceptualize multiple variables symbolically (Brainerd, 1978).

Ultimately, the philosophical basis on which this study is based is that the increased communication in a cooperative learning environment will spur a qualitative change in thinking skills of students and move them slowly forward into a higher plane of logical ability. In Vygotskian terms, the communication necessary in a cooperative learning environment will "scaffold" learners through their zones of proximal development (ZPDs) and move them forward in their cognitive development (Vygotsky, [1930s] 1978). Figure 1.1 introduces a graphical representation of this process.

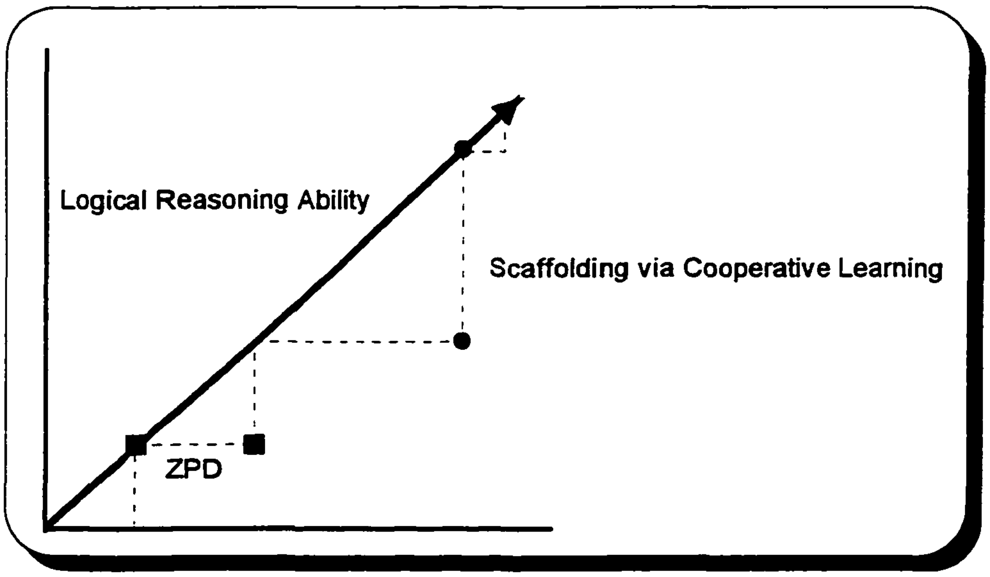

Logical Reasoning Ability
Scaffolding via Cooperative Learning
ZPD

Figure 1.1 Graphical Representation of Cooperative Learning and the Resulting Increase in Logical Reasoning Ability

It is expected that the cooperative learning environment will foster higher-order thinking skills necessary to move students through cognitive stages and hence improve their logical thinking skills. This research question will be examined with the null hypothesis $H_0$2.

$H_0$2: There will be no difference between the cooperative learning and control groups in the improvement of logical thinking skills.

Attendance

- Will a cooperative learning environment produce better attendance than a traditional lecture course?

Not only does the lecture pedagogy promote a feeling of anonymity among students, it fosters the feeling that it unnecessary for the student to attend. As feelings of isolation and alienation lead the student to believe that no one cares about his or her academic progress, a student may opt to give up and stop attending class. The more difficult the material becomes and the more pressure the student feels, the more important it becomes for the educator to facilitate social support in the learning situation (Johnson, et al., 1991). A cooperative learning environment provides an academic and social support system (Johnson & Johnson, 1994; Slavin, 1990). Therefore, it is expected that the cooperative learning environment will improve attendance. This research question will be examined with the null hypothesis $H_0$3.

H_03: There will be no difference between the cooperative learning and control groups in attendance.

Importance of the Study

The majority of research in cooperative learning has been conducted at the elementary and secondary levels. Interestingly, cooperative learning has not been heavily researched at the college level and particularly not in computer science (Johnson, et al., 1991). As late as 1993, Mehta claims to have performed the *first* empirical investigation of cooperative learning in a college computer science course. Her results were encouraging, finding a significant difference in test achievement between the control and experimental groups of students ($p \leq .01$). This study will attempt to reproduce her results regarding achievement in a more controlled manner with validated instrumentation.

In the last few years, other non-experimental research using cooperative learning at the college level has been conducted in computer science with generally positive results (Tenenberg, 1995; Walker, 1997; Yerion & Rinehart, 1995). Students report higher satisfaction with the cooperative learning courses and teachers feel that the material is learned at a "deeper level." None of these studies has been conducted simultaneously with a control group not using a cooperative learning environment. Any qualitative or observational data collected in this study will offer additional perspective and objectivity.

Achievement

Cooperative learning is still a relatively new paradigm in higher education, and especially in computer science. Most would agree that "group work" is a positive step, but merely forming groups and assigning work does not create a cooperative learning environment (Johnson, Johnson, & Smith, 1991). Significance may be claimed any time an increase can be produced in student achievement by a new teaching method.

Logical Reasoning Ability

The ramifications of a link between cooperative learning at the college level and an increase in logical thinking skills would be very profound. Recent constructivist learning theory suggests that a more active learning style enhances a student's learning potential. Vygotsky ([1934] 1962) contends that the interrelationships between thought and language are a key to consciousness and therefore imperative to developmental changes. Thought and language are not two separate events, Vygotsky maintained; the meaning in the thought is contained in the language and verbal communication enhances the thought process. His model of how learning happens hinges on small steps forward, which are "scaffolded" by other more knowledgeable peers, typically through verbal or written communication. Cooperative learning, by its very nature, causes an increased amount of communication. Most of this communication is verbal and the act of forming the words to explain a concept is a very important step in truly learning the concept.

Previous research has shown a link between course achievement in computer science and logical reasoning skills (Almstrum, 1994; Hudak & Anderson, 1990). If cooperative learning can be shown to help advance a student into the next operational stage, any discipline where these types of reasoning skills are necessary should consider cooperative learning as a potential educational teaching strategy.

Definition of Terms

Learning environment — a place (either physical or virtual) where a community of learners share a common set of psychological, pedagogical, cultural, technological, and pragmatic foundations (Land & Hannafin, 1996).

Logical reasoning — the ability to draw a logical inference or conclusion based upon the given facts.

High-level reasoning — the ability to evaluate and generate arguments in accordance with the principles of deductive and inductive inference (Nickerson, Perkins, & Smith, 1985).

Content comprehension — the capacity for the full understanding of specific computer science concepts.

Cognitive stages — as defined by Piaget: the sensorimotor, preoperational, concrete-operational, and formal-operational stages.

Mental stage transition — the movement by an individual from the current cognitive stage to the next (as ordered by Piaget).

Attendance — the number of times a person attends class.

Summary

This dissertation investigates the effects of cooperative learning on students enrolled in a second-semester university computer science course and, in particular, its effects on logical reasoning ability and achievement. The dissertation organization is as follows:

Chapter 1 serves as an introductory unit and provides a broad overview of the study.

Chapter 2 contains a review of the related literature. Philosophical and theoretical underpinnings for the research are identified, defined, and refined.

Chapter 3 describes the research design and methodology used in this investigation. Sampling procedures, instrumentation, and data analysis procedures are described and relevant literature within the domain of this methodology is included.

Chapter 4 presents the results of the investigation. The study's hypotheses previously presented in chapters 1 and 3 will be tested with the data gathered by the methods outlined in chapter 3. Results will be presented.

Chapter 5 concludes the dissertation with a discussion and interpretation of the study results. Conclusions and implications of the study are discussed, and suggestions for further research are offered.

Chapter 2. Review of the Literature

This chapter provides an overview and synthesis of the literature supporting this study. The review begins with an introduction to the process of learning, including its theoretical foundations and pedagogy in computer science. The review then continues with a discussion of the cooperative learning paradigm in general and, specifically, cooperative learning in university-level computer science together with reviewed research studies.

Theoretical Foundations

The process of learning is a well-studied and often argued area. Many theorists have claimed to understand exactly how humans learn, only to have their theories washed away in the next tide of research. This investigator posits that no single theory is uniquely correct in all situations, and therefore it is our job as educational researchers to fit theory and specific content together to form our own understandings of how learning actually occurs. Within this framework, it is acceptable—and possibly preferable—to selectively adapt portions of many different specific learning theories to form our own working theories. The two theorists most significant to this study are Jean Piaget and Lev Vygotsky.

Jean Piaget

The result of an early career in biology and a great interest in philosophy, Piaget's approach to psychology differs somewhat from those of his contemporaries. From the start, many of Piaget's theories were based upon the importance of biological structure and the idea that environment played a rather small part in the cognitive development process (Gallagher & Reid, 1981). His approach to studying cognitive function and his vision of solving philosophical controversies led him to label himself a *genetic epistomologist* (Brainerd, 1978), a label that reflects his biological view of cognition and learning.

Piaget posited that intellectual development is the result of the interaction of three components: cognitive structure, cognitive function, and cognitive content. A *cognitive structure* is the form that cognition takes, an abstract organizational pattern which controls cognition. In Piaget's theory, each stage of cognitive development is controlled by a set of unique cognitive structures. While cognitive structures underlie an individual's cognition, the *cognitive functions* are the goals for which the individual is striving during this ongoing cognitive development. *Cognitive contents* are the skills or abilities that comprise intelligence at any particular level (e. g. mathematical ability, reasoning skills, or abstract symbol manipulation) (Brainerd, 1978; Gallagher & Reid, 1981). It should be noted that in this theory, only cognitive contents can actually be directly measured.

Piagetian intellectual development occurs as the result of the growth of new cognitive structures, which can be used to promote cognitive function and store cognitive content. This growth is primarily *qualitative* because the structure acquires new functionality and the ability to process more difficult information. Proposing that this growth was primarily a process of maturation, Piaget developed his now-famous model of developmental stages—sensorimotor, preoperational, concrete-operational, and formal operational—and the age ranges by which time entry into each stage should be completed (Brainerd, 1978).

This investigation is primarily concerned with the fourth of these stages—the formal operational stage. This stage, which Piaget claims should be completed by age 15, is characterized by the ability to conduct hypothetico-deductive reasoning, scientific (inductive) reasoning, the ability for reflective abstraction. The term "formal" in formal-operational is derived from Piaget's claims that unlike the concrete-operational stage, mental operations may be executed from start to finish at a purely symbolic level. The cognitive content produced by the ability to use symbolic manipulation can be tested by an individual's ability to perform propositional logic (Brainerd, 1978).

Lev Vygotsky

Lev Vygotsky's writings have been a cornerstone in contemporary American psychology since the first translations of *Thought and Language* (Vygotsky, [1934] 1962). His socio-cultural theories have subsequently influenced educational theorists and curriculum development. Further, he is one of the founders of the constructivist movement (Bruner, 1962; Jaramillo, 1996), a movement founded on the idea that teachers must create experiences for students which provide them the opportunity to construct knowledge meaningfully in an appropriate social situation (Roth, 1990).

Vygotsky believed that society and the language used within society were crucial in the intellectual development of humans (Vygotsky, [1930s] 1978). One of the key elements in his theory of how humans learn is a construct he called the *zone of proximal development* (ZPD). Vygotsky defined the ZPD as:

> The distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under guidance or in collaboration with more capable peers. (Vygotsky, [1930s] 1978, p. 86)

In his theory, almost all learning is derived from contact with others. This causes a process of development that is less continuous than Piaget's maturation process, resulting in a rate of development that could even be described as in "leaps and spurts." A child's initial acquisition of language instigates one of these leaps. Unlike Piaget, who claimed that words merely represented meaning, Vygotsky contended that the meaning in thought is contained in the language. He further posited that the interrelationships between thought and language are key to consciousness and imperative to developmental change (Vygotsky, [1934] 1962). His mechanism for this developmental change is the ZPD.

The Vygotskian learning model hinges on communication between the learner and more knowledgeable peers or adults who help or "scaffold" the learner through the ZPD and cause the learner to internalize this new-found skill or knowledge. This internalization is the process by which Vygotsky claimed that cognitive functionality advances (Vygotsky, [1930s] 1978).

How does Vygotsky's theory manifest itself in a practical sense? One key is to actively engage students in solving problems that they feel are genuinely problematic (Roth, 1990). Through this engagement cognitive change can occur through human interaction and communication embedded in meaningful day-to-day activity (Vygotsky, [1934] 1962, [1930s] 1978). Face-to-face interaction allows students to "try out" the social and academic discourse necessary for them to become active researchers and creators of their own knowledge (Wertsch, 1991).

Piaget and Vygotsky differed in their basic theories of cognitive advancement (Piaget, 1962; Vygotsky, [1934] 1962; [1930s] 1978). Piaget's stages are defined on a "macro" level—he posits only four stages for an entire lifetime. Vygotsky, in contrast, proposed a continual series of "micro" levels, in which a person is continuously moving forward through one ZPD to the next level of cognitive ability. This study combines the two perspectives by claiming the constructivist framework of Vygotsky can be described in Piagetian terms. Instruments calibrated to determine Piaget's levels will be used to measure advancement in logical reasoning created through the normal constructivist learning process.

Pedagogy and Introductory Computer Science

Technology changes rapidly and computer science pedagogy must change with the times. Most computer science curriculums require students to possess skills in abstraction much earlier in the computer science curriculum than in earlier years (Dale & Lilly, 1995), in keeping with the recommendations of the ACM-IEEE-CS Joint Curriculum Task Force's *Computing Curricula 1991* (Tucker, 1991), which encourages the simultaneous use of theory, abstraction, and design.

The task force's description of abstraction stresses data collection and hypothesis formulation, modeling and prediction, experimental design, and result analysis as crucial skills for future computer scientists (Tucker, 1991). The ability to hypothesize about data structures and how algorithms will modify or use these structures is essential for success in computer science, an ability that is becoming especially important as more complicated software-creation methodologies arise, such as object-oriented programming (Guzdial, 1995).

The Role of Logic in the Curriculum

Computing Curricula 1991 also emphasizes the role of logic in the computer science curriculum. Its recommendations suggest that a course in discrete mathematics should be included early in the curriculum so that its concepts may be applied throughout the remainder of a student's coursework. Among the topics included in the typical discrete mathematics course include "sets, functions, elementary propositional and predicate logic, Boolean algebra, elementary graph theory, matrices, proof techniques (including induction and contradiction), combinatorics, probability, and random numbers" (Tucker, 1991, p. 27).

Although this movement toward logic early in the curriculum seems appropriate, it may actually present a problem for many students, as a number of recent studies show. Almstrum (1994) found that novice computer science students experienced more difficulty with concepts involving mathematical logic than they did with more general computer science concepts ($p \leq .002$). She further found that individual performance is strongly correlated ($p \leq .01$) to the examination questions involving logical relationships.

Kim (1995) concluded that propositional reasoning ability is related to course achievement in a logic class for computer science majors. She also found that a logic course specifically designed to teach the logical concepts measured in her study did *not* improve the students' performance. One of Kim's recommendations for further research is to study new ways to teach introductory computer science courses, especially the logic courses.

One interpretation of the previous findings is that the students had not progressed far enough through Piaget's cognitive stages to understand the logic necessary for success in computer science. This theory is supported by the fact that students' skills in logic did not improve even though they were specifically being taught logic concepts. In contrast, Piaget's explanation for these phenomena was that the peak of intelligence is reached at adolescence and as we grow older, we lose cognitive ability. This theory is now widely discounted (Brainerd, 1978).

In a direct application of Piaget's theories on hypothetical reasoning, Hudak and Anderson (1990) found that success (as measured by achieving 80% or above in the course) was highly related to formal operational ability ($p \leq .01$). This finding bolsters the results of Barker and Unger (1983) and Kurtz (1980) who also found a positive relationship between formal operational status and success in computer science courses. An important notion in Hudak and Anderson's findings was that an inability to abstract and grasp the logic involved, along with the simultaneous use of multiple variables was key to the failure of students who had not yet reached the formal operational stage (Hudak & Anderson, 1990).

Other Computing Curriculum 1991 Recommendations

Computing Curriculum 1991 is a comprehensive guide designed to also address various non-content issues. Two of the "Other Educational Experiences" (Tucker, 1991, p. 20) listed were "working as a team" and "communication." Students need to develop the communication skills, both oral and written, necessary to function well in these team environments. A primary goal of the "team" concept in learning situations is to add realism to the curriculum. As software professionals, most students will be involved in large software projects where teamwork is essential.

The success or failure of these recommendations can be tied directly to the pedagogy used to deliver the essential content. One of the task force's suggestions is to include a mandatory laboratory component to the curriculum along with the lecture. Each of these laboratory sessions is to be well planned and the completion of the lab should be accompanied by a written report (Tucker, 1991). Obviously, the traditional lecture/discussion/laboratory paradigm drove these recommendations.

Learning Paradigms

Although many of the world's activities are conducted cooperatively, the pedagogy used in higher education is quite often competitive or individualistic (Johnson, Johnson, & Smith, 1991). Teachers teach as they were taught (typically using a traditional lecture format), which makes change in academia a slow process. Although still not the norm, strategies for cooperative learning have been successfully used at the college level and are constantly challenging the current lecture-based learning paradigm (Purdom & Kromrey, 1995).

The Lecture Method

The *lecture*, an extended presentation in which the instructor presents factual information in an organized and logical way, is currently the most commonly used paradigm in college teaching (Johnson, et al., 1991). A lecture is easily adaptable to different audiences and the amount of time it consumes is easily controlled. The lecture's prevalence in contemporary pedagogy does have some basis in educational psychology.

The rationale for the pedagogy of lecturing is based upon some very basic theories of the structure and organization of knowledge such as Bruner's (1960) theory of knowledge structures and categorization of information. Using such a structure, new information is tied to previously existing structures through the use of advance organizers that should be present in a well-organized lecture (Ausubel, 1963). Through the use of the lecture large amounts of information can be disseminated in a short period of time. If the lecturer is organized and entertaining, the lecture can be a very useful tool.

The main problem with the lecture pedagogy is that the learner is typically not active. Due to this passivity, attention suffers and the material in the lecture is missed or remembered incorrectly (Penner, 1984). McKeachie and Kulik (1975) found that lecturing was effective for the transmission of factual knowledge, but discussion was a superior means of promoting higher-level reasoning and problem solving.

The Cooperative Learning Method

As explained in chapter 1, student-student interactions can be classified as *competitive,* where one student's success depends upon another's failure, as *individualistic,* where no direct competition for grades exists, but the students work by themselves with little or no collaboration, or as *cooperative* where members of a group work toward a common goal and are committed to maximize the learning potential of each member of the group. Researchers have taken many approaches to building cooperative learning environments.

Student Teams-Achievement Divisions (STAD)

In a STAD (Slavin, 1978) cooperative learning environment, students are assigned to heterogeneous four-member learning groups. After the teacher presents a lesson, the students work within their groups to make sure each member has mastered the lesson. Students take individual quizzes individually with scoring based upon past performance. Team rewards are given to motivate the students to help each other. The two primary emphasized principles in a STAD environment are positive interdependence and individual accountability.

Teams-Games-Tournaments (TGT)

Teams-Games-Tournaments (DeVries & Slavin, 1978) uses the same basic structure as STAD, but substitutes weekly tournaments for the quizzes. Students compete for points against other students of similar abilities. Winners contribute points to their respective teams, but compete without teammates, which assures individual accountability.

Jigsaw

The Jigsaw (Aronson, Blaney, Stephan, Sikes, & Snapp, 1978) environment uses six-member teams to work on academic material that has been broken down in such a manner that no individual can solve the problem without the help of the other group members. Individuals become "experts" in one area of the problem and meet with other experts in that same area from different cooperative groups. Evaluation works in the same manner as the STAD model, with individual quizzes and team rewards.

Group Investigation

Groups Investigation (Sharan & Sharan, 1976) is a less formal cooperative learning method. Students form groups of two-to-six persons and plan small research projects that are then presented to the class. Each group is responsible for a portion of a unit that is being studied by the entire class.

Learning Together

Learning Together (Johnson & Johnson, 1994) methods are much like those of the STAD environment. Students form heterogeneous groups of three-to-five members and work on course materials. The basic characteristics of a successful Learning Together cooperative learning program are: positive interdependence, individual and group accountability, promotive interaction, interpersonal skills, and group processing (Johnson & Johnson, 1994).

- **Positive Interdependance**—Each group member must contribute to the success or failure of the group and each member must realize that this relationship exists. Positive interdependence creates a situation where each member must learn the material and also help other group members who don not fully understand that material. The group must know that it "sinks or swims" together.
- **Individual Accountability**—Although the members must act as a group, human nature is such that each member must have an identity and have personal responsibility. Ultimately, the purpose of the group is not only for the group to succeed by attaining the group goals, but for each member to become a stronger individual.
- **Promotive Interaction**—The members of a group must promote each others' successes. This notion is the antithesis of a competitive learning situation and a key to the success of cooperative learning. Face-to-face promotive interaction is best because it helps develop a more mature relationship among the group members.
- **Interpersonal Skills**—Small-group skills help the students interact in a more productive manner. The skills needed to learn academic subject matter

and to function as a group are often referred to as *teamwork* (Slavin, 1985). As in an athletic situation, it is important for team members to learn to trust each other, communicate accurately, support each other, and resolve conflicts constructively.

- **Group Processing**—The tasks necessary for the completion of a project must be completed as a group and not as individuals. It is important that the group members reflect upon what is successful and unsuccessful in terms of achieving the group's goal(s). The group must also make decisions regarding the continuation or termination of a group's activities. The quality of the group's taskwork and teamwork should continually improve over the course of the collaborative effort.

Each of these models has attributes that make it desirable for a particular age group. In the context of this investigation, "Learning Together" appears to be the most applicable because it is a more general set of guidelines and not necessarily targeted for elementary-age students. The STAD evaluation techniques are primarily driven by past performance. In a university course evaluation situation, the grading has to be a bit more normalized and cannot be dependent on improvement only.

Research on the Cooperative Learning Paradigm

Throughout the years, cooperative learning has been one of the most investigated learning paradigms. Since 1898 over 400 research studies have been conducted regarding cooperative, competitive, and individualistic learning (Johnson & Johnson, 1994). These studies have led to the confirmation of cooperative learning's effectiveness on both a theoretical and practical level. In general, students in the cooperative learning groups showed higher achievement (2/3 of a standard deviation higher than the individualistic groups). Cooperative learning groups also showed more higher-level reasoning, more frequent generation of new ideas, and greater knowledge transfer between subjects (Gabbert, et al., 1986; Johnson & Johnson, 1994; Skon, et al., 1981).

Positive Aspects of Cooperative Learning

Cooperative learning can offer some advantages not found in the more traditional learning environments. Several are described below.

Active Learning

Bruner (1960) recommends the formation of a learning environment marked by *disequilibrium*—a state caused by the introduction of a new concept which is in conflict with a prior schema or misconception (Brainerd, 1978). This type of environment spurs learning by keeping the student involved and active in the educational discussion. A cooperative learning environment is ideal for attaining this goal. Students challenge each other to look at material from different points of view and to reevaluate their positions, strengthening meta-cognitive skills.

On a practical note, college students report higher satisfaction in courses that encourage group discussion (Bligh, 1972; Kulik & Kulik, 1979). This satisfaction promotes a more positive attitude toward the subject area and a continuing motivation to learn (Johnson & Johnson, 1994). Active engagement with the material and a positive classroom environment—one in which students are provided an opportunity to construct meaningful knowledge—encourages a student to become involved and practice the logic of discovery (Roth, 1990).

Peer Teaching

From a social constructivist view, learning is an activity that occurs between people and not between a person and inanimate objects or ideas. When students collaborate with other students and faculty members, they join a community which tests the basis of ideas and reality (Whitman, 1988). Student-student collaboration can be an especially powerful tool for creating such a community in a non-threatening manner (Whitman, 1988).

The Roman philosopher Seneca said, "Qui Docet Discet" (to teach is to learn twice). As teachers we can identify with this statement. Small groups give students an opportunity to explain concepts to each other. Within this small-group environment, a student who is more knowledgeable about a subject will take on the role of the "scaffolder." In this context he/she is known as a *peer tutor* (Johnson & Johnson, 1994). Both the peer tutor and the rest of the group benefit from this situation. The tutor "relearns" the material and also gets practice teaching. Although peer teachers may be a bit clumsy at first, with reinforcement for effective tutoring they can become effective communicators (Johnson & Johnson, 1994). The rest of the group benefits by hearing the material again and most likely presented in a different manner.

Higher-Order Thinking

One of the primary goals of science education is to develop individuals who can evaluate information and make reasoned hypotheses and judgments. In short, to develop individuals "who can sort sense from nonsense" (Johnson & Johnson, 1994, p. 57). Ruggerio (1988) posits that the primary means of teaching critical thinking is not *what* is taught, but *how* it is taught. He concludes, "the only significant change that is required is a change in teaching methodology" (p. 12). Studies have shown that cooperative learning promotes a greater use of higher-level reasoning and critical thinking than do competitive or individualistic learning environments (Gabbert, et al., 1986; Skon, et al., 1981).

Cooperative learning environments provide more opportunity for student elaboration and active engagement with the concepts via student discussion. This in turn allows the student to integrate new information with prior knowledge and view these concepts from different perspectives (Johnson & Johnson, 1994). The cooperative learning process also encourages the verbalization of methods and strategies that develop metacognitive skills (McKeachie, 1988).

Interpersonal Skills

Because many of the most crucial skills necessary for success in the workplace rely on the ability to work well with others, interpersonal skills may be as valuable as technical skills. Software engineers commonly work on projects that are much too large for any one person to complete. The ability to communicate functionality, interface design, and detailed instructions is crucial. These skills are fundamentally different than those needed to actually program a computer, a process that is often isolating and very self-absorbed (much like the process of writing a dissertation). Lew, Mesch, Johnson, and Johnson (1986a, 1986b) found that socially isolated students learned more about social skills and engaged in them more frequently in a cooperative learning situation than in an individualistic situation.

Collaboration Within Cooperative Learning

One key to the success of cooperative learning is the collaboration which takes place among the members of a group (Johnson & Johnson, 1994). This collaboration can take many forms, but generally falls into the categories of cognitive and motivational collaboration (Slavin, 1990).

In the cognitive domain, Vygotsky ([1930s] 1978) described the influence of collaboration on learning by describing how functions are first created in the group in the form of relations among students and then become mental functions for the individual. Cooperative learning is also cognitively effective because students in the same class are likely to be operating within one anothers' zones of proximal development (Slavin, 1990). At some point during the learning process, it is likely that each student in the cooperative learning group will serve as the "more capable peer" who helps another student through his or her zone. Cooperative goal structures create situations where the only path to individual goals is through the successful completion of group goals. Further, group members will encourage groupmates to exert a maximum effort. In other words, students are expected to pull their weight and work hard toward the group goals.

Motivational factors also contribute heavily to the success of cooperative learning environments (Slavin, 1990). Because cooperative goal structures create situations in which the only path to individual goals is through the successful completion of group goals, group members will encourage each other to exert a maximum effort. In other words, students will expect each other to pull their own weight and work hard toward the group goals.

Johnson and Johnson (1985) describe patterns of social interdependence and types of relationships that form more often among students in a cooperative learning experience than in a competitive or individualistic experience. They found that cooperative learning promoted a greater degree of interpersonal attraction and more positive relationships between groups of homogenous students, students from different ethnic groups, and handicapped and nonhandicapped students. Perhaps the most tangible outcome of these positive relationships is the increase in self-esteem reported by students within cooperative learning environments (Slavin, 1990). Two factors cited as important in the increased self-esteem are academic improvement and a feeling that students are well-liked by their peers.

Cooperative Learning in University-Level Computer Science

A pressing need in computer science education is to discover the types of learning environments that will be effective in our rapidly changing technological world. To this end, researchers have begun to study the use of cooperative learning in university computer science courses, with generally positive results.

Quite often computer science is thought of as a solitary profession; with such a thought, individualistic teaching makes sense. This perspective is changing as it has become more apparent that cooperative learning may better mirror the team-oriented workplace that students will find after graduation (Sabin & Sabin, 1994). Because of this, cooperative learning is becoming more accepted in the university computer science classroom.

Mehta (1993) claims to have performed the first empirical research in cooperative learning in a university computer science course. Over the course of three semesters, she collected data in her introductory computer programming course (CS 1). The first semester—taught in the traditional lecture/discussion style—served as a control group for the following two semesters that used a modified Jigsaw cooperative learning strategy.

Her results regarding achievement were encouraging, yielding significant differences between the cooperative learning classes and the control group on several exams, but not on the overall testing average for the semester. The study itself had some limitations including a very small sample size (a total of N=31 over three semesters), and a lack of validated instrumentation. Although these limitations make reliance on her results problematic, her study did employ valid cooperative learning techniques and can be used as a stepping stone to further research.

In a similar study, Sabin and Sabin (1994) report a significant pretest/posttest improvement for a cooperative learning group (t-test, $p \leq .01$) using a modified STAD technique. The investigators claim that the achievement levels of the two groups were similar because the posttest means are similar (see Table 2.1), but it appears that no attempt was made to compare the experimental and control groups on the variable of achievement by using a statistical technique such as an Analysis of Covariance (ANCOVA). The researchers also report positive effects—as measured by a 20-item agree/disagree questionnaire—on course satisfaction, classroom atmosphere, the proper use of computer science terminology, and student demeanor.

Table 2.1 Reported Pretest and Posttest Means from Sabin and Sabin (1994)

Class	Control (N=18)	Experimental (N=13)
Pretest Mean (5=highest)	3.90	2.70
Posttest Mean	4.30	4.20

The study by Sabin and Sabin (1994) is promising, but also problematic for reliability. The small sample size is further confounded by the fact that the control and experimental groups were drawn from different populations measured both by major and gender: The control group was primarily computer science majors with a male/female ratio of 16:2, while the experimental group was primarily mathematics majors with a male/female ratio of 5:8. As in the Mehta (1993) study, none of the instrumentation was validated.

In a non-experimental study, Tenenberg (1995) reports high student evaluation and positive attitudes as a result of a Johnson and Johnson (1994) based cooperative learning environment. He also made some interesting observations regarding the move toward a cooperative learning environment: Because most students have been conditioned to expect a lecture/discussion format, and have developed successful strategies within this setting, some students resisted the change to a new paradigm. This student resistance was also found to be a disruptive influence for Yerion and Rinehart (1995), who also conducted a non-experimental study of cooperative learning in a first-semester university computer science course. Tenenberg reported that knowing the literature and success rates of cooperative learning environments, as well as listening to students' concerns, can alleviate many of the problems regarding student resistance. Tenenberg also discussed the changing role of the educator. Cooperative learning entails "stepping down off the podium," a radical change in teaching style; the new role of facilitating group work, acting as a consultant, and leading class discussions was very new and different for the instructor as well as the students. Lastly, Tenenberg noted that he was not able to cover as much material because it took slightly longer to cover the planned topics. This was viewed both positively and negatively as he felt that the topics covered were discussed at a deeper level. Cooperative learning environments do not necessarily cover less material, however, Walker (1997) reports that his classes cover 10%-20% more material than traditional lecture/discussion sections without a loss of detail.

Undergraduate Student Attendance

Common sense would dictate that a student's continued presence in class would be beneficial to that student. It is difficult for a student to be "on task" in the classroom when not even present. Further, student attendance contributes to superior student learning, classroom management, and ultimately meeting legal and professional responsibilities (Petress, 1996).

A study designed to determine the distinguishing factors between high and low achievers in an undergraduate biology class found attendance to be very important (Nist, Holschuh, & Sharman, 1995). Van Blerkom (1992) found a significant correlation ($p \leq .03$) between attendance and examination scores in 17 sections of an undergraduate psychology class. He further reported that attendance waned throughout the semester partly because the students felt discouraged and disengaged.

Unlike a traditional competitive classroom, a cooperative classroom creates an environment where a student who works hard, shows up for class, and helps others is praised and encouraged by groupmates (Slavin, 1990). Previous studies in cooperative learning have shown that a cooperative learning environment positively affects students' attitude and time on task (Johnson & Johnson, 1989; Slavin, 1990). Klein and Pridemore (1992) investigated the effects of cooperative learning and the need for group affiliation on-time on-task at the university level. Students showed a significantly higher amount of time on task in a cooperative environment ($p \leq .001$).

Summary

The computer science curriculum's move toward added abstraction and discrete mathematics early in students' coursework when they may not yet have developed the necessary cognitive complexity, necessitates a close look at current computer science pedagogy. Cooperative learning environments have had generally positive results in university computer science classes (Mehta, 1993; Sabin & Sabin, 1994; Tenenberg, 1995; Walker, 1997; Yerion & Rinehart, 1995). With only two experimental studies in the area, the need exists for further examinations of the benefits of cooperative learning in computer science education.

This study builds on the existing research to answer the following questions already introduced in chapter 1:

- Will students in a cooperative learning environment comprehend computer science content better than students in a traditional lecture course?
- Will cooperative learning create an environment which helps students move through Piaget's cognitive stages and improve their logical thinking skills?
- Will a cooperative learning environment produce better attendance than a traditional lecture course?

Chapter 3 describes the methodology by which the research questions developed in chapters 1 and 2 will be explored.

Chapter 3. Methodology

Many research efforts in cooperative learning have concluded that a cooperative learning environment enhances the learning process and results in higher achievement (Aronson, et al., 1978; Johnson & Johnson, 1989; Slavin, 1990). Unfortunately, most of these studies have been conducted using participants who have not yet reached the university level (Johnson et al., 1991). Further, very little objective evidence has been collected to determine the effects (if any) of a cooperative learning environment in a university computer science classroom (Mehta, 1993). This study investigated the effects of cooperative learning on content comprehension, logical reasoning, and attendance in a second-semester computer science class.

The hypotheses in this investigation were tested by exposing the treatment group to a cooperative learning environment. The control group was given no treatment; it received instruction with a traditional lecture/discussion format. This investigation is quasi-experimental in form (due to non-random sampling) and employs a pretest-posttest control group design (Campbell & Stanley, 1963).

This chapter describes the steps undertaken to test the hypotheses and answer the research questions posed in this investigation. The chapter begins with a detailed description of the study so that it can be reproduced for the purpose of scientific validity, including descriptions of the population sample, dependent and independent variables, instrumentation, and data collection methods. Secondly, the treatment is described in precise detail so that critical evaluations may be made of the techniques used in the investigation. Finally, the methods of statistical analyses used will be described in order to clarify results presented in chapter 4.

Research Hypotheses

The following hypotheses were tested in this investigation:

$H_0$1: There will be no difference between the cooperative learning and control groups in concept comprehension.

$H_0$2: There will be no difference between the cooperative learning and control groups in the improvement of logical thinking skills.

$H_0$3: There will be no difference between the cooperative learning and control groups in attendance.

Population Sample

The population sample for the investigation consisted of all students enrolled in Computer Science 315 (CS 315) in a large southwestern research university during the summer term of 1996. CS 315 (the equivalent of CS 2 at many universities) is a required course for computer science majors at this university. The course is typically taken as the second course in the computer science sequence. Students must have a grade of C or better in the prerequisite CS 304P (or equivalent) to enroll in CS 315.

The students in this sample had all passed a first-semester course in computer science (CS 1) with Pascal programming (or an equivalent; Tucker, 1991) and were considered representative of the population of second-semester computer science students at the university level. Several participants had previously taken CS 315 and had not passed (or passed with a D, which is not sufficient for computer science degree credit). These students were not excluded from the study because any truly representative population is likely to have subjects of this type.

The total number of students who enrolled in the two course sections was N=67. Students were assigned to either the control or experimental group by their choice of course time via the normal course registration procedure. The morning section served as the control for the afternoon experimental section. Both sections of the course were taught by this researcher. This type of sample is a "convenience sample" because of the lack of randomization in choosing second-semester computer science students from the entire population of colleges and universities.

Control Group

The control group consisted of the students in the morning section of the CS 2 course, which met three days a week for 90 minutes of class each day, and one day a week for a 90-minute discussion (a total of 360 minutes per week). The control group was conducted using the traditional lecture/discussion method. Students in the control group were responsible for doing all programming and other assignments by themselves in an individualistic mode. The control group was given six programming assignments, three quizzes, two midterm exams, and a final examination over a period of 9 weeks.

Treatment Group

The treatment group consisted of the students in the afternoon section of the CS 2 course, which met three days a week for 90 minutes of class each day, and one day a week for a 90-minute discussion. The treatment group employed a cooperative learning environment for all class instruction based upon the principles of cooperative learning described in Johnson, Johnson, and Smith (1991). The treatment group's discussion section was organized in the same fashion as that of the control group. The treatment group was given similar programming assignments and the same quizzes and examinations as the control group over the same period of 9 weeks.

Final Sample After Attrition

During the first week of the investigation, a colleague collected the following data from the students in both sections: a consent form (see Appendix A), a demographics questionnaire (see Appendix B), a pretest for the Propositional Logic Test (PLT) (see Appendix E), and a pretest for the Burton Comprehension Instrument (BCI) (see Appendix C). The posttest for the PLT (see Appendix F) and the posttest for the BCI (embedded in the final exam, see Appendix D) were administered during the last week of the semester.

Of the 67 students who registered for the courses, 49 completed every data collecting instrument and signed the consent form (23 in the morning section and 24 in the afternoon section). Some students were absent on one or both days the data collecting instruments were given. Therefore, the portions of the investigation that used the BCI or PLT were conducted with a final total of N=49; the attendance investigation used the entire class (67 students over N=23 lectures).

Dependent Variables

This investigation had three dependent variables: comprehension, logical reasoning, and attendance.

Comprehension

Comprehension is defined as the capacity for understanding fully specific computer science concepts. Comprehension was measured using the Burton Comprehension Instrument (BCI) (Burton, 1992). This instrument consists of 25 multiple choice questions derived from the Educational Testing Service's (ETS) Advanced Placement examination in computer science. The instrument was designed to measure the comprehension of five critical topics in the CS 2 course: complexity, stacks, queues, recursion, and sorting. The BCI was administered by giving the pretest portion of the BCI the first week of class and inserting the prescribed test items into the final examination.

Logical Reasoning

Logical reasoning is defined as the ability to draw a logical inference or conclusion based upon the given facts. The instrument utilized for measuring students' logical reasoning skills was the Propositional Logic Test (PLT) (Piburn, 1985). The PLT was administered during the first week of class and the alternate PLT (A-PLT) was given during the last week of class to both groups.

Attendance

Attendance is defined as the number of times a person attends class. Attendance was taken by a head count at the beginning of each class period. Students were not graded on their attendance or participation.

Independent Variable

This investigation had one independent variable: the students' type of learning environment.

Learning Environment

Learning environment is defined as a place (either physical or virtual) where a community of learners share a common set of psychological, pedagogical, cultural, technological, and pragmatic foundations (Land & Hannafin, 1996). There were two learning environments in this investigation: a traditional lecture/discussion environment and a cooperative learning environment based upon the works of Johnson and Johnson (1994).

Methodology

As described previously, two sections of CS 315 were formed during the summer of 1996 at a major southwestern research university for the purpose of investigating the effects of cooperative learning on second-semester university computer science students. In addition to the methods described below, frequent instructor-student and student-student contact was made via e-mail and course materials were distributed via the World Wide Web (WWW) to students in both sections.

The main emphasis in CS 315 is the abstract data type (ADT), a collection of data values and operations. Hence, the focus of the course is the specification, implementation, and application of the ADT. Because students in CS 315 are relative beginners in computer programming, proper software engineering principles such as modularity, data encapsulation, information hiding, and the proper design of algorithms are also stressed in the course (Dale & Lilly, 1995).

Turbo Pascal 6.0 from Borland, Inc., was chosen as the language to be used for class programming assignments. This choice was made for several reasons: the students' own familiarity with the Pascal programming language from the CS 304P course; the fact that the language supports (but does not necessarily enforce) the desired software engineering principles; and the fact that the university computer science department supports this platform in its general-purpose computing laboratories.

Both the control and treatment groups had access to the same laboratory facilities on campus. These laboratories were quite accessible, but the choice of the MS-DOS platform for program implementation led to a large number of students opting to use their home personal computers for coursework. Although this effect was purely unintentional, extensive home-computer use had the desirable effect of reducing the contact between the control and cooperative learning groups.

Each section had a teaching assistant who was responsible for grading programs and quizzes, delivering a short lecture, holding office hours, and answering students' questions regarding programming assignments. Teaching assistants and the investigator met several times a week to make sure the same material was being covered in the discussion sections and to confer on grading policy.

The remaining discussion of the investigation's methodology describes the learning environments of the control and cooperative learning sections. A "typical day" in each type of classroom is described and differences in the environments are shown.

The Lecture/Discussion Environment

A typical 90-minute class period in the lecture/discussion (control) section of the experiment had the following format. The first five minutes were devoted to questions from the class. These questions were generally about the lecture from the previous day or questions regarding an upcoming programming assignment. Following the initial question/answer session, a lecture of approximately 80 minutes was presented. Students were encouraged to ask questions or provide comments throughout the lecture. At times "volunteers" were chosen to come to the board and trace through examples or solve problems. This technique proved useful for keeping students' attention throughout the class. The lecture was often supplemented with overhead transparencies provided as part of the course materials by the authors of the textbook (Dale & Lilly, 1995). The last five minutes of each class period were reserved for questions. In general, the control group of the investigation was taught in much the same fashion as most introductory computer science classes at the university level.

The control group met for a discussion section once a week. All quizzes, exams, and programming assignments were administered in the discussion section. The teaching assistants also presented brief lectures and held question/answer sessions during the discussion sections. Most of the questions in the discussion sections concerned the programming assignments.

The Cooperative Learning Environment

The cooperative learning model used in this study was an adaptation of that described by Johnson and Johnson (1994). Importantly, the basic elements of a cooperative learning environment (positive interdependence, individual accountability, promotive interaction, interpersonal skills, and group processing) as described in the previous chapter were integrated into the design of the course through the grading criteria and group formation.

All students in the treatment section were simultaneously assigned to two types of groups: Heterogeneous *base groups* consisted of four or five students selected on the basis of demographic data collected during the first class session. Base groups remained intact throughout the entire term, and were designed to provide academic and social support.

Most of the actual in-class coursework was completed in *working groups*, each of which consisted of three students. Working groups were formed prior to each of the term's three examinations and dissolved after each examination. Each student was therefore assigned to a total of three successive working groups over the course of the semester. To maximize the number of relationships formed, students were assigned to working groups heterogeneously with respect to base group membership and prior working group membership; no student was placed in a working group with another student from a previous working group.

Each working group completed one or two programming assignments and handed in one set of homework problems before each examination. All students in a given working group received identical grades on the programming assignments and homework; each student was individually responsible for his/her own test grade.

Getting Started

The first day of class is always important for setting a positive tone in any classroom setting. The investigator was especially aware of the potential for resistance to the cooperative learning methodology and grading. For this reason, a large part of the first class period in the cooperative learning class was spent explaining exactly what was going to happen in the class, making it clear that although work would be completed in groups, each individual would ultimately be responsible for his or her own grade. An effort was made to make students feel at ease and to assure the students that this was not an unusual method for teaching this course. An especially effective argument for teaching a programming course cooperatively was to explain that in the "real world" programming is rarely an individual task. At no time did the investigator want the students to feel that they were part of a study conducted by their own instructor. This would have seriously compromised any data gathered. For this reason, Dr. Nell Dale administered the pretests and gathered the consent forms for the investigation under the guise that the research was being conducted by the computer science education research group and not by the investigator.

Two principal aspects of the cooperative learning model are positive interdependence and individual accountability. The grading structure of the course was designed to help enforce these concepts. Students took all examinations and quizzes by themselves, thereby making each student individually accountable. In order to create a positive interdependence, bonus points were given to base groups in which all the students passed or in which each student did better than his or her last examination or quiz.

Homework and programming assignments were completed in the working groups. Each group handed in one set of homework before an exam and one or two program assignments. Each member of the group got the same grade on the homework and the program(s). After each working group had been disbanded, its former members were given the opportunity to confidentially examine the group dynamics (see Appendix G) and relate any important information as to the amount of effort put forth by each group member. Students are very aware of what is "fair" in a classroom and it is important to reassure the students that their individual work will be rewarded.

This process of self- and group evaluation is very important and is vital to meeting the criteria for group evaluation as set forth in the chosen cooperative learning model. The "one grade per assignment" was essential in establishing positive interdependence.

On the second day of class, a colleague administered the pretest instruments and a questionnaire. Several items on the questionnaire were specifically designed to measure past performance and the likelihood of success in this course. These questions included grade-point average (GPA), last computer science course and the grade earned in that course, last mathematics course and the grade earned in that course, and the number of hours of work outside of school (see Appendix B). Answers to these items allowed the investigator to place the students into one of three classifications: high performer, average performer, and below-average performer. The purpose of this classification was to better create heterogeneous groups based upon past performance, which in the investigator's experience has been a strong predictor of future performance. The primary indicators used in this categorization were overall GPA and participants' grades in previous computer science courses.

At the beginning of each semester, a certain amount of fluctuation exists in any class roster. Students drop and add courses because of scheduling conflicts and other academic and personal needs. After the third day of class, the investigator assigned students to base groups by reviewing the information on the questionnaires and choosing students from each of the three previous-performance classifications for each group. The groups were created to be heterogeneous based upon expected performance. The investigator also chose the first set of working groups at this time.

The "Typical" Day

In a typical class for the experimental group, students began class by meeting in their base groups for five minutes to discuss any problems or other issues before them. Next, students broke into working groups to work on problem sets covering material for the next examination. (Programming assignments were completed outside of class.) After about 20 minutes, the instructor presented a short (10-15 minute) lecture on a particularly important or difficult topic. After that, working groups reconvened and worked until the end of the class period—and often well after.

Cooperative Environment Validation

Although an accurate portrait of the learning environment created for this study can be gathered from the above description, the investigator invited external reviewers to validate his conception of a cooperative learning environment.

Several educational experts were invited to attend the class to make observations and to answer the following question: "Does the learning environment you observed correspond to your definition of a cooperative learning environment?"

The results of this review were quite interesting because the reviewers came from different backgrounds and brought their own perceptions of cooperative learning to the discussion. Each reviewer agreed that the environment did correspond to his definition of a cooperative learning environment. (For the full reviews, see Appendix H.)

Instrumentation

Burton Comprehension Instrument

The Burton Comprehension Instrument (BCI) was developed specifically to measure comprehension of five key topics in a CS 2 course: complexity, stacks, queues, recursion, and sorting (Burton, 1992). The BCI contains 25 questions in a pre- and posttest format that measure concept comprehension. These questions were gathered from the Education Testing Service's (ETS) advanced placement (AP) examinations for the years of 1984, 1988, and 1990.

Structure

The BCI pretest and posttest contain 25 multiple-choice items with the following distribution:

- five questions on complexity
- nine questions on stacks and queues
- five questions on recursion
- six questions on sorting

Participants were given 15 minutes to complete the BCI pretest. The posttest was given by embedding the items into the final exam. (See Appendix D.)

Reliability

The reliability of a test is its internal consistency and ability to measure accurately and consistently upon each administration. The overall Cronbach Alpha reliability of the questions in the BCI is 0.76 (Burton, 1992).

Validity

The BCI is a domain-specific instrument that was specifically developed for measuring the five content areas of CS 2 described above. A test of validity for such an instrument hinges on whether the instrumentation accurately examines the content area. The items in the BCI were first determined to be valid by ETS and later by Burton and the author of the textbook used in the CS 2 course used in this investigation (Burton, 1992).

Propositional Logic Test

The Propositional Logic Test (PLT) was developed by a group of researchers at Rutgers University for the purpose of measuring propositional operations and conditional reasoning. The design of the PLT was guided by the formal rules of logic and its purpose is to measure the participant's ability to interpret truth-functional operators by identifying instances that are consistent or inconsistent with a stated rule (Piburn, 1989, p. 667).

Structure

The PLT has 16 items which can be further broken into four 4-item subtests. Each subtest is designed to address a Piagetian operation. Each item on the PLT is a propositional statement (in binary form) followed by four choices which are accepted or eliminated by the statement. The test takes approximately 15 minutes to administer and the participants may refer to the directions at any time (see Appendix E).

Reliability

The reliability studies of the PLT have been very consistent. Enyeart (1980) reported a 0.90 reliability on a sample of university students. Kim (1995) noted that in her pilot study a Cronbach Alpha reliability of 0.85 was achieved. Her study was conducted at the same university as this investigation with a population of computer science students that can be considered very similar to the participants in this study.

Validity

The validity of the PLT was measured by correlating PLT results with other measures of logical reasoning. Of special interest were the correlations with longer, in-depth clinical analyses. The correlations from the other measures of logical reasoning range from 0.31 to 0.78 (Piburn, 1985). A measure of content validity was performed by a panel of four logicians who agreed that the PLT reflected the logical meaning of the operators in question and that it was a valid instrument for measuring a participant's logical reasoning ability (Piburn, 1990).

Alternate PLT

The original PLT was designed as a posttest-only instrument. As a part of her study, Kim found it necessary to create (by randomizing the original questions) an alternate version of the PLT (A-PLT, see Appendix F) for use in a pretest-posttest design. Cronbach's Alpha reliability on the A-PLT was determined to be 0.87 (Kim, 1995).

Other Data Gathering

Questionnaire

A brief questionnaire was developed by the investigator to collect demographic information from the participants, such as last mathematics course taken and grade earned, last computer science course and grade earned, GPA, and hours worked outside of school (see Appendix B).

Attendance

A head count was taken during each class period to determine the number of students in attendance (see Appendix I). Discussion-section attendance was not counted because both group's discussion sections were taught in a traditional manner and were not therefore a part of the independent variable.

Group Evaluations

Each student was required to submit an evaluation of his or her group's performance upon the completion of the required assignments (see Appendix G).

Course Evaluations

Course evaluations were administered during the last week of the course in accordance with university policy.

Data Analysis

Comprehension

$H_0$1: There will be no difference between the cooperative learning and control groups in concept comprehension.

- The analysis for the comprehension hypothesis was an Analysis of Covariance (ANCOVA) with the pretest for the BCI as the covariate.

Logical Reasoning

$H_0$2: There will be no difference between the cooperative learning and control groups in the improvement of logical thinking skills.

- The analysis for the logical reasoning hypothesis was a Multivariate Analysis of Variance (MANOVA) for repeated measures.

Attendance

$H_0$3: There will be no difference between the cooperative learning and control groups in attendance.

- The analysis for the attendance hypothesis was a one-way Analysis of Variance (ANOVA).

The SPSS statistical analysis program for Windows 95 was used to analyze the data and calculate the results of the statistical analysis. An alpha level of 0.05 was used to measure significant differences between the groups. This alpha level was chosen because of the preliminary nature of this investigation. Because this is one of the few investigations of its type, the more stringent alpha level of 0.01 may result in the premature closure of a research avenue. If significant results are found at the suggested alpha level, subsequent research can verify the results with more stringent levels of significance.

Limitations of the Investigation

As with all research in the social sciences, controlling for the individual differences of the participants was very difficult. Differences in anxiety, motivation, or attitude may play a part in the success or failure of an individual student. With the limited time and resources available, it was impossible to account for every possible contributor to within-group variation.

Another limitation of this study was the inability to determine if the effects of the treatment were due to the cooperative learning or the increased active learning. Active learning is a major component of a cooperative learning environment. Distinguishing which part of the treatment (the cooperation or active learning) caused the differences was difficult, but not critical to the results because cooperative groups encompass active learning.

The study had a relatively small sample size because it was conducted during a summer term. Although the study was being conducted at a large research university, the results are possibly more generalizable to a smaller university or college with smaller class sizes. The sample size also has negative ramifications on the power of the study. Statistically significant differences in populations are much more difficult to find as the sample size decreases.

Summary

This chapter described the research methods and design for the investigation. Most importantly, the cooperative learning environment, as implemented in this study, was described in detail for critical review. The hypotheses, variables, treatment, data collection techniques, and data analysis were presented along with supporting information on the instrumentations' reliability and validity. Several limitations of the study were then outlined. The results of the study will now be presented in chapter 4.

Chapter 4. Results

This investigation examined the effects of cooperative learning on concept comprehension, logical reasoning, and attendance of students enrolled in a second-semester university computer science course. The experimental design, methodology, instrumentation, and data analysis techniques for this investigation were described in chapter 3. This chapter presents the results of the data analysis.

Overview of Data Analysis

Population Sample

The two sections of a second-semester computer science class (CS 315) at a major southwest research university contained a total of 67 students during the summer session of 1996. Of these students, 49 completed the consent form and questionnaire, took the pretest and posttest for the BCI, and took the pretest and posttest for the PLT. Therefore, the number of subjects for the statistical evaluations is 49 (N=49).

These students met for a total of 23 class sessions during the summer term (not counting discussion sessions). Therefore, the number of data points for the test of the hypothesis regarding attendance is N=23.

The statistical procedures used in the analysis of the data were an Analysis of Covariance (ANCOVA), a Multivariate Analysis of Variance (MANOVA) for Repeated Measures, and a one-way Analysis of Variance (ANOVA). Each of these tests was performed with the Statistical Package for the Social Sciences (SPSS) on a Windows 95 platform.

Results of Data Analysis

Concept Comprehension

Will students in a cooperative learning environment comprehend computer science content better than students in a traditional lecture course? In the last 90 years, nearly 400 studies have been conducted to determine if cooperative learning promotes higher productivity and achievement. A meta-analysis of these studies shows that in general students score roughly two-thirds of a standard deviation higher in a cooperative learning environment than in either a competitive or individualistic environment (Johnson & Johnson, 1989). Therefore, it was expected that the students in the cooperative learning environment would comprehend the material better and show greater achievement than the students in the traditional lecture. This research question was examined with the null hypothesis $H_0 1$.

$H_0$1: There will be no difference between the cooperative learning and control groups in concept comprehension.

Although Analysis of Covariance (ANCOVA) accounts for individual differences from the pretest to the posttest, it is prudent to determine if both samples are from the same population. Table 4.1 contains the means and standard deviations on the BCI of the lecture and cooperative learning groups. The means between groups on the BCI pretest are not significantly different (F=.731, p=.397); therefore, the samples/subjects are indeed from the same population.

Table 4.1 Means and Standard Deviations for the BCI Pretest and Posttest

Group	N	BCI Pretest Mean	BCI Pretest Standard Deviation	BCI Posttest Mean	BCI Posttest Standard Deviation
Lecture	25	7.44	3.54	18.36	3.49
Cooperative Learning	24	8.29	3.43	18.86	3.48

A test for difference on the BCI posttest between the two sections was conducted using an ANCOVA. Table 4.2 presents the ANCOVA results using the BCI pretest as the only covariate. One standard procedure when using an ANCOVA is to test for the homogeneity of group regressions. This test was run to determine if any interaction took place between the covariate and the treatment. The test for homogeneity of group regressions was not found to be significant (F=.000, p=.989). The assumption was therefore met and the analysis of covariance could be continued. There was no significant difference found between the lecture and cooperative learning group in the area of concept comprehension (F= 0.530, p=.471). Therefore, null hypothesis $H_0$1 was not rejected.

Table 4.2 ANCOVA Result Comparing the Cooperative Learning and Control Groups on the Burton Comprehension Instrument (BCI)

Source of Variation	SS	df	MS	F	Significance of F (p)
Within Cells	478.83	46	10.41		
By Class	5.51	1	5.51	0.530	0.471

Logical Reasoning

The hypothesis undergirding this research is that increased communication in the cooperative learning environment will spur a qualitative change in the thinking skills of the students and move them slowly forward toward a higher plane of logical ability. Therefore it was expected that the cooperative learning environment would foster higher-order thinking skills necessary to move students through cognitive stages and hence improve their logical thinking skills. This research question was examined with null hypothesis $H_0$2.

$H_0$2: There will be no difference between the cooperative learning and control groups in the improvement of logical thinking skills.

Although the Multivariate Analysis of Variance for Repeated Measures (MANCOVA) accounts for the individual difference from the pretest to the posttest (as with an ANCOVA), it is likewise prudent to determine if samples are from the same population. Table 4.3 contains the means and standard deviations of the lecture and cooperative learning groups on the Propositional Logic Test (PLT). The means between the groups on the PLT pretest are not significantly different (F=3.036, p=.088), so it was concluded that on this measure, too, the samples were indeed from the same population.

Table 4.3 Means and Standard Deviations for the PLT Pretest and Posttest

Group	N	PLT Pretest Mean	PLT Pretest Standard Deviation	PLT Posttest Mean	PLT Posttest Standard Deviation
Lecture	25	11.12	4.43	13.20	3.79
Cooperative Learning	24	13.17	3.75	14.17	2.68

A test for difference between the cooperative learning and control groups on the PLT was conducted using a MANOVA for Repeated Measures. The repeated measures over time consisted of the pretest and posttest for the PLT. Table 4.4 shows the summary results of the MANOVA. No significant difference was found between the lecture and cooperative learning groups on the improvement in logical reasoning as measured by the PLT (F=2.00, p=.164). Therefore, null hypothesis $H_0 2$ was not rejected.

Table 4.4 MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the Propositional Logic Test (PLT)

Source of Variation	SS	df	MS	F	Significance of F (p)
Residual	167.92	47	3.57		
Class by Time	7.14	1	7.14	2.000	0.164

The PLT consists of four 4-item subtests. Each subtest addresses one of the Piagetian operators: conjunction, disjunction, implication, or biconditional. Additional information regarding students' logical reasoning ability can sometimes be gained by looking at the subscale test scores for theses areas. Table 4.5 shows the group means for each of the pretest and posttest subtests.

Table 4.5 Means for the Pretest and Posttest PLT Subtests

Group	N	Conjunction	Disjunction	Implication	Biconditional
Lecture	25	3.56/3.80	3.04/3.40	2.48/2.96	2.16/3.04
Cooperative Learning	24	3.66/3.42	3.38/3.54	3.00/3.42	3.13/3.79

Tables 4.6 through 4.9 show the results of the MANOVA for repeated measures on each of the PLT subtests. As with the overall measure of significance on the PLT, the results of a MANOVA for repeated measures show no significant differences between the lecture/discussion and cooperative learning groups on any individual subtest.

Table 4.6 MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Conjunction Subtest

Source of Variation	SS	df	MS	F	Significance of F (p)
Residual	37.53	47	0.80		
Class by Time	1.47	1	1.47	1.840	0.181

Table 4.7 MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Disjunction Subtest

Source of Variation	SS	df	MS	F	Significance of F (p)
Residual	27.55	47	0.59		
Class by Time	0.23	1	0.23	0.390	0.535

Table 4.8 MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Implication Subtest

Source of Variation	SS	df	MS	F	Significance of F (p)
Residual	31.04	47	0.66		
Class by Time	0.02	1	0.02	0.040	0.848

Table 4.9 MANOVA for Repeated Measures Result Comparing the Cooperative Learning and Control Groups on the PLT Biconditional Subtest

Source of Variation	SS	df	MS	F	Significance of F (p)
Residual	46.99	47	1.00		
Class by Time	0.28	1	0.28	0.280	0.600

Attendance

Will a cooperative learning environment be associated with better attendance than that of a traditional lecture course? Cooperative learning creates a large amount of social interdependence and strong peer relationships. Studies have shown that cooperative learning environments are likely to produce lower attrition rates in individual classes, especially if used throughout the college experience (Wales & Sager, 1978). Therefore it was expected that the cooperative learning group would have higher attendance than the traditional lecture group. This question was examined using null hypothesis $H_0 3$.

> **$H_0 3$: There will be no difference between the cooperative learning and control groups in attendance.**

Because each course section had a different number of students, comparing the raw head count from each class would not be an accurate measurement for means of comparison between classes. For this reason, the percentage of students enrolled in the class on a given day was used as the data point for comparison between classes. The fact that the number of students enrolled in a class varied during the semester because of withdrawals and drops was taken into consideration when calculating the daily attendance percentages (see Appendix I).

A test for difference in the attendance between the cooperative learning and control groups was conducted using an Analysis of Variance (ANOVA). Table

4.10 shows the daily mean attendance and Table 4.11 shows the results of the comparison. A significant difference was found at the alpha level of .05 between the two groups (F=5.054, p=.030). Therefore, the null hypothesis $H_0 3$ was rejected.

Table 4.10 Means and Standard Deviations for Attendance (N=23 lectures)

Group	N	Attendance Mean	Attendance Standard Deviation
Lecture	23	0.83	0.11
Cooperative Learning	23	0.90	0.09

Table 4.11 ANOVA Results on the Attendance Between Classes (N=23 lectures)

Source of Variation	SS	df	MS	F	Significance of F (p)
Residual	0.365	44	0.008		
Between Classes	0.042	1	0.042	5.054	0.030

Summary of Findings

In the course of this investigation, three hypotheses were tested. Table 4.12 contains a summary of these hypotheses and results.

To summarize, no statistically significant difference was found between the lecture and cooperative learning sections on the measure of content comprehension (F=.530, p=.471). Therefore the first hypothesis was not rejected. In addition, no significant difference between the lecture and cooperative learning groups was found on the measure of logical reasoning (F=2.00, p=0.16). Therefore the second hypothesis was not rejected.

A statistically significant difference was found between the lecture and cooperative learning sections on the overall class attendance variable (F=5.054, p=.030) at the .05 level of significance. Therefore, the third hypothesis was rejected.

Table 4.12 Summary of Hypothesis Testing

Hypothesis	Result
$H_0 1$: There will be no difference between the cooperative learning and control groups in concept comprehension.	Fail to Reject
$H_0 2$: There will be no difference between the cooperative learning and control groups in the improvement of logical thinking skills.	Fail to Reject
$H_0 3$: There will be no difference between the cooperative learning and control groups in attendance.	Rejected

Chapter 5. Discussion

This chapter discusses how the results of the investigation reported in this study may be integrated into the existing theory and research on cooperative learning in university-level computer science courses. It begins with a summary of the investigation. Next is an examination of the significance, implications, and limitations of each research question. The chapter concludes with recommendations for further research.

Summary

Attrition rates in the computer science major are quite high. Many students who struggle through the first few courses ultimately drop out of the major when the coursework becomes too complex, mostly because of the increased amount of logic required for the courses. Could an attempt to create an atmosphere which fosters an improvement in logical reasoning skills improve the success rate for many computer science students? Furthermore, will this improvement lead to better content comprehension?

This study compared content comprehension, logical reasoning ability, and attendance in two groups of second-semester university computer science students. In a quasi-experimental, pretest/posttest, control-group design, the control group (n=25) received instruction in a traditional lecture/discussion learning environment three days a week for nine weeks. The treatment group (n=24) met in a cooperative learning environment (as defined by Johnson and Johnson, 1994) for the same number of hours as the control group. Each group was given the pretest and posttest for the Burton Comprehension Instrument (BCI) (Burton, 1992) and a pretest and posttest for the Propositional Logic Test (PLT) (Piburn, 1985) to measure levels of content comprehension and logical reasoning ability. A head count was taken daily to determine if the cooperative learning environment might promote better attendance.

Results and Discussion

Content Comprehension

Will students in a cooperative learning environment comprehend computer science content better than students in a traditional lecture course?

Most studies have shown that cooperative learning environments promote higher achievement than traditional educational environments (Johnson & Johnson, 1989; Slavin, 1990). The results of hypothesis testing for this investigation revealed that on the BCI, students in the cooperative learning section did no better than the lecture/discussion group ($p \leq 0.47$), an unexpected result, somewhat contradicted by midterm and semester grades. As shown in

Table 5.1, the cooperative learning section did moderately better than the lecture/discussion section on the first midterm exam ($p \leq 0.072$). On the second midterm exam, the cooperative learning section did significantly better ($p \leq 0.005$) than the lecture/discussion group. It must be noted that these two examinations are instruments that have not been validated. Nor have the scores been adjusted for prior knowledge.

Table 5.1 Means and Standard Deviations for the Midterm Examinations

Group	N	BCI Pretest Mean	Exam 1 Standard Deviation	Exam 2 Mean	Exam 2 Standard Deviation
Lecture	25	108.44	12.83	114.48	10.81
Cooperative Learning	24	114.71	10.83	*123.00	9.07

* Significant at the alpha=0.01 level.

The significant difference found in the midsemester examination scores corroborates the findings of Mehta (1993), who also found that her cooperative learning sections did significantly better than her lecture sections. Walker (1997) notes that the students in his cooperative groups learned the material at a "deeper" level than his past students.

So why did the BCI not detect any significant differences? First, when used in a pretest/posttest fashion using an ANCOVA, the prior knowledge of the cooperative group is added into the ANOVA by adjusting the means. This had some effect on the scores, but the raw posttest scores on the PLT were not significantly different to begin with ($p \leq 0.31$). A more likely cause is that the instrument is not sensitive enough to detect the differences in content comprehension. Although previously used in a pilot study and investigation of closed laboratory effectiveness in a second-semester university computer science course (Burton, 1992) and validated through that process, a more precise instrument may be necessary to detect true content comprehension in this population.

Logical Reasoning

Will cooperative learning create an environment that helps students move through Piaget's cognitive stages and thus improve their logical thinking skills?

McKeachie (1988) concludes that three of the most important activities that can increase students' thinking skills are student discussion, explicit emphasis on problem solving using varied methods and examples, and verbalization of methods and strategies to encourage the development of metacognition. These activities are all central to a cooperative learning environment. Indeed, studies have shown that cooperative learning promotes a greater use of higher-level reasoning strategies and critical thinking than competitive or individualistic learning (Gabbert, et al., 1986; Skon, et al., 1981; Slavin, 1985).

The results of this investigation do not support an increase in logical reasoning ability through a cooperative learning environment. The results of hypothesis testing regarding logical reasoning ability via the PLT showed no significant difference between the control group and the cooperative learning group. Any discussion of these results must be prefaced by the comment that detecting a change in logical reasoning ability through an intervention is a very difficult task. Table 5.2 shows the pre- and posttest means for the PLT. Note the relatively high score of the cooperative group on the pretest (13.17 of a possible 16).

Table 5.2 Pretest and Posttest Means on the Propositional Logic Test (PLT)

Group	N	PLT pretest Mean	PLT pretest Standard Deviation	PLT posttest Mean	PLT posttest Standard Deviation
Lecture	25	11.12	4.43	13.20	3.79
Cooperative Learning	24	13.17	3.75	14.17	2.68

Additional information regarding the patterns of student mistakes can sometimes be found through the analysis of the subscale tests on the PLT. An analysis of the subscale test scores for conjunction, disjunction, implication, and biconditional reasoning revealed no significant differences and scores within expected ranges. The only anomaly was a slight decrease in score on the conjunction subtest for the cooperative learning group. This is somewhat surprising because the conjunction subtest is considered to be the easiest of the four. The investigator can posit no explanation for this decrease.

The cooperative learning group's mean score on the pretest was almost as high as the lecture/discussion group's posttest score. This created an unfortunate situation in which (because of the sampling via normal course registration) the treatment group was marginally different than the control group ($p \leq 0.088$) and had very little room to grow in ways measurable by the PLT, which ranges in score from 0-16.

Another ramification of this difference in logical reasoning ability is that this very difference may have contributed to the significant differences in content comprehension. Previous research has linked success in computer programming courses with logical reasoning ability (Barker & Unger, 1983; Hudak & Anderson, 1990; Kurtz, 1980). It appears likely that the individuals who comprised the cooperative learning section had an advantage in that they began the course with a higher level of logical reasoning ability. Although statistically the two groups were sampled from the same population, an advantage nevertheless appears to have existed prior to the treatment.

Attendance

Will a cooperative learning environment produce better attendance than a traditional lecture course?

The results of this study show that a cooperative learning environment positively affects attendance in a second-semester computer science course ($p \leq .03$). This finding is not surprising due to the increased amount of peer interaction and the need for the entire group to work together on tasks. A student is *expected* to show up for class by the other members of the group and missed if not there, thus forming a type of *collaborative community* (Blumenfeld, et al., 1996).

Unlike a traditional competitive classroom, a cooperative classroom creates an environment where a student who works hard, shows up for class, and helps others is praised and encouraged by groupmates (Slavin, 1990). Previous studies in cooperative learning have shown that a cooperative learning environment positively affects student's attitude and time on task (Johnson & Johnson, 1989; Slavin, 1990). Klein and Pridemore (1992) investigated the effects of cooperative learning and the need for group affiliation on-time on-task at the university level. Students showed a significantly higher amount of time on task in a cooperative environment ($p \leq .001$).

Qualitative Observations

In addition to the collection of quantitative data, qualitative data was gathered throughout the investigation in the form of external reviewer comments (see Appendix H) and feedback from the students via group evaluation forms (see Appendix G). The data gathered via these instruments will not be rigorously examined in this dissertation, but trends and interesting observations will be commented on to further describe qualitative differences between the cooperative learning and lecture/discussion groups.

Students in the cooperative learning class were required to submit group evaluation forms after each examination for the purpose of providing additional feedback for the grading process. The most useful comments resulted from the following two statements: "List at least one positive aspect of your group's performance" and "How could the group have performed better?"

Aspects of Positive Group Performance

One of the key premises of this investigation was that increased communication would enhance logical reasoning ability. Another hope was that critical thinking skills would be utilized more frequently during the cooperative learning experience. The investigator was pleased to discover that many students reported that the expression of varying opinions and the careful examination of these opinions were crucial to the group learning process and to solving various programming and homework problems. Other students reported that "learning from others" was a positive aspect of their groups' performances. Other positive aspects reported included: good communication, finishing assignments, sharing the workload, friendship, and cooperation.

Reports of the groups' positive aspects changed slightly over the course of the nine weeks and three working groups. The first working groups' set of evaluations contained more responses regarding multiple opinions than subsequent evaluations. Perhaps the students were initially surprised at this newfound freedom to solicit other's opinions, but then took it for granted later in the semester. The subsequent two sets of group evaluation tended to stress coordination, teamwork, and other more practical computer programming skills such as similar coding style, modularity, and testing ability.

Group Improvement

Unlike comments about the positive aspects of the working groups, the comments regarding how the groups could improve changed radically over the course of the semester. The first working groups had difficulty in the area of work distribution. Most commonly a single group member dominated the work and did not let others contribute to their full extent. The investigator attributes this to an initial lack of trust and an unfamiliarity with working in groups. Subsequent group evaluations did not mention the distribution of work as a problem, implying that the groups learned how to distribute the work load more effectively—a very important skill.

After the first working group, the most common complaints involved a lack of meeting time outside of class, time management, or group members not attending class or meetings. Students became more involved with other coursework as the semester progressed. Many students listed "start earlier" as an indication of how their group could have performed better. Time management was a definite problem. Clearly students expected each other to attend class and participate in meetings. This expectation contributed to the excellent attendance shown in the class.

Conclusions

Based on the quantitative results of this study, cooperative learning appears to be no more effective than the traditional lecture/discussion environment as a pedagogy for teaching computer science content in a second-semester university classroom. Furthermore, the cooperative learning environment did not improve the students' logical reasoning ability when compared with a traditional lecture/discussion environment. However, the investigation did find that the cooperative learning environment produced significantly better attendance when compared to a traditional lecture/discussion environment.

Although the quantitative results are discouraging, the investigator recommends more research be conducted on cooperative learning methods in university computer science courses. The students in the cooperative learning section of the course did not perform significantly better than the students in the lecture course, but they also did no worse. In addition, they were exposed to many other positive learning situations that are not possible in a lecture environment. These situations and the skills learned from them—such as teamwork, cooperation, and conflict resolution—will prove to be valuable in future workplace situations.

Future Research

Although cooperative learning has been widely studied, more needs to be done—especially at the university level and especially in computer science. One potential area of study is in the effectiveness of cooperative learning in larger sections of classes. Most introductory courses at larger universities will continue to be taught in larger sections because of the lack of funds for more faculty. Some preliminary research is promising. Kleiner (1992) notes that the use of cooperative writing assignments made a significant difference in the students' level of critical thinking in an introductory psychology course. Recently, Mason (1997) showed that a small class of at-risk students did no better than a group of at-risk students who used cooperative techniques within a larger lecture setting. Due to the logistics involved, a complete cooperative learning environment, such as the one described in this study, may not be feasible with hundreds of students, but a hybrid or lecture/discussion and cooperative learning may yield positive results.

One of the limitations of this study was that the students in a second-semester computer science course had less variability in talent than would a first-semester class—the weaker CS students would not have passed the first course and hence not advanced. A suggestion for further research is that this study be replicated on the population of first-semester computer science students. The cooperative groups would be more heterogeneous because of the greater range of student talent levels, thereby providing more opportunity for scaffolding and student development.

Because of the overwhelming evidence that formal-operational reasoning ability is necessary for success in computer science (Almstrum, 1994; Barker & Unger, 1983; Hudak & Anderson, 1990; Kim, 1995; Kurtz, 1980) more research needs to be done on how to prepare future computer science students. This preparation is likely to be focused in the high schools. Perhaps changes in secondary science or mathematics can focus on instructional approaches that may produce positive effects on reasoning development (Almstrum, 1991).

Most research in cooperative learning takes one of two forms: (1) comparative investigations of student achievement or attitude (much like this investigation); and (2) research on how to apply cooperative learning techniques in the classroom (Tobin, Tippins, & Gallard, 1994). More research must be conducted on how cooperative learning works, and not merely on if it works or how to apply it in a given educational situation. One aspect of this research may involve measuring the forms of collaboration which occur within cooperative learning and the effects of this collaboration on individual's Zone of Proximal Development. For example, how do motivation, caring, consensus building, and effort affect learning within a cooperative learning environment?

Finally, in the two sections of the course described in this investigation 11 of the 67 students were females (4 in the cooperative learning section and 7 in the lecture/discussion). Thus 16 percent of the class was female. This number is similar to the 13 percent reported in Kim's (1995) study. Further, of these 11 female students, only 2 were not of a minority group. Park (1993) found that females in a cooperative learning section did significantly better in an introductory chemistry class than females who worked individually. Walker (1997) claims that females perform better in a cooperative learning environment and in Sabin and Sabin's (1995) study, the experimental section consisting of 8 females and 5 males made a significant pretest-to-posttest improvement. No study has yet targeted cooperative learning and gender performance in the area of university computer science.

Appendix A

Consent Form

Consent Form

You are invited to participate in a study measuring content comprehension and logical reasoning ability in second semester computer science students. There will be approximately 70 students from The University of Texas Computer Sciences Department in this study. Permission is also being asked to follow your progress through university records in further computer science courses to determine the significance of the findings.

If you are willing to participate, **there are no specific tasks you must perform outside the normal requirements of your computer science courses.** Any information that is obtained from you or about you in connection with this study **will remain strictly confidential.** We do not foresee any reason to disclose data from any individual because we are only interested in group data.

Your decision whether or not to participate will not prejudice your future relations with The University of Texas at Austin. If you decide to participate, you are free to discontinue participation at any time without prejudice.

If you have any questions, please ask now. If you have any additional questions later, Professor R. Priebe (471-9737) or Professor N. Dale (471-9539) will be happy to answer them.

You will be offered a copy of this form to keep.

Your signature indicates that you have read the above information and have chosen to participate.

________________________________ ________________

signature date

____

please print your name here

____

signature of investigator

Appendix B

Questionnaire

Questionnaire

Name ____________________

Course __________________

What was your last computer programming class?

What grade did you earn?

What was the last math course you took?

High School ___

Pre-Calculus ___

Calculus I ___

Beyond Calculus I ___

What grade did you earn?

What is your overall GPA?

Are you working outside of school?

If so, how many hours per week?

Thank you.

Appendix C

Burton Comprehension Instrument (BCI) Pretest

CS 315 Preliminary Test

Please fill in the letter of the BEST answer to each of the following questions on the scantron.

1. Consider the following four tasks:
   1. To perform a linear search of a list of n names
   2. To perform a binary search of a sorted list of n names
   3. To perform a selection sort into alphabetical order of a list of n names that are initially in random order
   4. To perform a merge sort into alphabetical order of a list of n names that are initially in random order

For large n, which of the following lists these tasks in order (from least to greatest) of their worst-case running times?
(a) 2, 4, 3, 1 (b) 4, 3, 1, 2 (c) 2, 4, 1, 3
(d) 2, 1, 4, 3 (e) 2, 3, 1, 4

2. If N is the number of data elements to be manipulated by one or more algorithms, then the execution time of each such algorithm can be characterized in terms of N using the Big-O notation. Of the following, which best characterizes an execution time that is significantly different from the others?
(a) O(N) (b) O(2N) (c) O(N + 2)
(d) O(N/2) (e) O(N)

3. An ADT where the retrieval operation returns the item that has been in the structure the least amount of time.
(a) list (b) stack (c) FIFO queue
(d) priority queue (e) circle

4. An ADT where the retrieval operation returns the item that has been in the structure the longest time.
(a) list (b) stack (c) FIFO queue
(d) priority queue (e) circle

5. Of the following data representations for storing integers with an arbitrary number of digits, which would allow two integers having that representation to be added and the result to be stored using the same representation in the most time-efficient manner?
(a) A binary search tree with the nodes containing the digits and their positions in the decimal representation of the integer and with the nodes ordered by the digits
(b) A stack of digits, with the leftmost (most significant) digit on top of the stack
(c) A stack of digits, with the rightmost (least significant) digit on top of the stack
(d) A linked list of digits, proceeding from the leftmost (most significant) digit to the rightmost (least significant) digit
(e) A linked list of digits, proceeding from the rightmost (least significant) digit to the leftmost (most significant) digit

6. function z(k,n:integer):integer;

```
begin
if n = k then
    z := k
else
    if n > k then
        z := z(k,n-k)
    else
        z := z(k-n,n)
end;
```
Based on the function defined above, what is the value of z(6,8)?
(a) 1 (b) 2 (c) 3 (d) 4 (e) 8

7. It would be most appropriate to use a recursive function or procedure to solve a problem that
(a) can be reduced to two, or more, simpler or smaller cases of the same problem
(b) involves a substantial number of conditionals and nested loops
(c) requires a lot of memory
(d) involves storing data in a two-dimensional array
(e) involves evaluation of the factorial function

8.
```
program Main;
   var z:integer;
   function F(x:integer):integer;
   begin
        if (x = 1) or (x = 3) then
             F := x
        else
             F := x * F(x - 1)
   end;
begin  (* main *)
z := F(F(2) + F(5))
end.
```
If MAXINT were large enough to allow the program above to be executed, then at the end of the program, the value of z would be
(a) 62   (b) 5! + 2!   (c) (5! + 2!).!
(d) (7!)!   (e) (62!)/(2!)

9. Merge sort (internal) has which of the following advantages over bubble sort for long lists?

i. Merge sort requires much less coding to implement than does bubble sort
ii. Merge sort runs faster than bubble sort
iii. Merge sort requires less storage space than bubble sort

(a) i only   (b) ii only   (c) iii only
(d) i and iii (e) ii and iii

10. How many comparisons are required to sort an array of length 5 if a straight selection sort is used?
(a) 5   (b) 10   (c) 15   (d) 20   (e) 25

11. How many comparisons are required to sort an array of length 5 if a straight selection sort is used and the array is already sorted?
(a) 0 (b) 1 (c) 10 (d) 20 (e) 30

12. How many comparisons are required to sort an array of length 5 if the bubble sort with a Boolean flag, ShortBubble, is used?
(a) 5 (b) 10 (c) 15 (d) 20 (e) 25

13. How many comparisons are required to sort an array of length 5 if ShortBubble is used and the array is already sorted?
(a) 0 (b) 1 (c) 4 (d) 5 (e) 10

14. In QuickSort, if the splitting value is the first element in the array, then QuickSort will be most efficient with input data in:
(a) sorted order (b) random order (c) nearly sorted order
(d) reverse order (e) law and order

Questions 15 and 16 deal with the following problem. The height of a binary tree is the number of nodes in the longest path from the root to a leaf of the tree. The height of an empty tree is 0; the height of a single-node tree is 1.

```
1   Function Height(Tree:TreeType):integer;
2   Begin
3     If _______________ then    (* question 15 *)
4        Height := 0
5     else
6        __________________      (* question 16 *)
7   end;
```

15. Fill the blank in line 3.
(a) Tree <> NIL (b) Tree = NIL (c) Tree^ = NIL
(d) Tree^.next = NIL (e) none

16. Fill the blank in line 6.
(a) Height(Tree^.right) + Height(Tree^.left)
(b) Height(Tree^.right) + Height(Tree^.left) + 1
(c) Height := Height(Tree^.right) + Height(Tree^.left) + 1
(d) Height := Max(Height(Tree^.right), Height(Tree^.left)) + 1
(e) none

Questions 17 - 22 deal with the following problem.
You have a situation where you need to use two stacks, S1 and S2. You know that together they will never have more than Max elements. You decide to use an array representation with both stacks residing in the same array but with two logical pointers to top, Top1 (for S1) and Top2 (for S2). Notice this will not be a linked list. Use the following definitions and declarations.

```
CONST
    Max = 100;
    MaxPlus1 = 101;
TYPE
    ItemType = (* component on the stack *)
    StackType = record
                    Stack:array [1..Max] of ItemType;
                    Top1: 0 .. Max;
                    Top2: 1 .. MaxPlus1
                end;
```

17. Choose the statement that correctly implements this procedure.

```
Procedure ClearS2(Var S2:StackType);
Begin
    (* statement goes here *)
end;
```

a. S2.Top2 := 0;
b. S1.Top1 := 0;
c. S2.Top2 := MaxPlus1;
d. S2.Top2 := Max;
e. none

18. Choose the statement that correctly implements this function.

```
Function Full(S1,S2:StackType):boolean;
begin
    (* statement goes here *)
end;
```

a. Full := S1.Top1 = Max;
b. Full := S1.Top1 = S2.Top2;
c. Full := S2.Top2 - S1.Top1 = 1;
d. Full := S1.Top1 + S2.Top2 = MaxPlus1;
e. none

Questions 19 and 20 deal with the following procedure shell.

```
Procedure PushS1(Var S1:StackType; Item:ItemType);
Begin
    S1.Top1 := ____________________; (* question 19 *)
    ____________________ := Item  (* question 20 *)
end;
```

19. Choose the expression that makes the first statement correct.
(a) S1.Top1 + 1   (b) S1.Top1 - 1   (c) Max - S1.Top1
(d) S1.Top1 + MaxPlus1   (e) none

20. Choose the expression that makes the second statement correct.
(a) Stack[S1.Top1 + 1] (b) Stack[MaxPlus1 - S1.Top1]
(c) Stack[S1.Top1] (d) S1.Stack[S1.Top1] (e) none

Questions 21 and 22 deal with the following procedure shell.

```
Procedure PopS2(Var S2:StackType; Var Item:ItemType);
Begin
    Item := _______________;  (* question 21 *)
    S2.Top2 := _____________;  (* question 22 *)
end;
```

21. Choose the expression that makes the first statement correct.
(a) Stack[S2.Top2]   (b) Stack[Max - S2.Top2]
(c) S2.Stack[Top2]   (d) S2.Stack[S2.Top2]   (e) none

22. Choose the expression that makes the second statement correct.
(a) S2.Top2 + 1   (b) S2.Top2 - 1
(c) MaxPlus1 - Top2   (d) Max - S2.Top2   (e) none

Questions 23 through 25 use the following table.

N	column a	column b	column c
32	5	1024	160
64	6	4096	384
128	7	16384	896
256	8	65536	2048
512	9	262144	4608
1024	10	1048576	10240
2048	11	4194304	22528
4096	12	16777216	49152

You have run different algorithms to accomplish the same task and these were the values that were returned as *amount of work* for the appropriate value of N (first column).

23. Which of the following would most closely describe the complexity of the algorithm returning the values in column a?

(a) N (b) logN (c) NlogN (d) N^2 (e) 2^N

24. Which of the following would most closely describe the complexity of the algorithm returning the values in column b?

(a) N (b) logN (c) NlogN (d) N^2 (e) 2^N

25. Which of the following would most closely describe the complexity of the algorithm returning the values in column c?

(a) N (b) logN (c) NlogN (d) N^2 (e) 2^N

Appendix D

CS 315 Final Examination with Embedded BCI Posttest

CS 315 Final Exam

Name ____________________

TA ____________________

Part I

Circle the most correct answer. (2 pts each)

1. An assertion that states what is true before execution of a code segment.

   a. invariant   b. postcondition   c. precondition

   d. truth value   e. abstraction

2. An assertion that states what is true after execution of a code segment.

   a. invariant   b. postcondition   c. precondition

   d. truth value   e. abstraction

3. An assertion that is always true.

   a. invariant   b. postcondition   c. precondition

   d. truth value   e. abstraction

*4. An ADT where the retrieval operation returns the item that has been in the structure the least amount of time.

   a. list   b. stack   c. FIFO queue

   d. priority queue   e. circle

*5. An ADT where the retrieval operation returns the item that has been in the structure the longest time.

   a. list   b. stack   c. FIFO queue

   d. priority queue   e. circle

*6. How many comparisons are required to sort an array of length 5 if a straight selection sort is used?

   a. 5   b. 10   c. 15   d. 20   e. 25

*7. How many comparisons are required to sort an array of length 5 if a straight selections sort is used and the array is already sorted?

   a. 0   b. 1   c. 10   d. 20   e. 30

*8. How many comparisons are required to sort an array of length 5 if the bubble sort with a Boolean flag Switch is used?

   a. 0   b. 1   c. 10   d. 20   e. 30

*9. How many comparisons are required to sort an array of length 5 if the bubble sort with a Boolean flag Switch is used and the array is already sorted?

   a. 1   b. 4   c. 10   d. 20   e. 30

10. Testing based on data coverage.

a. black box b regression c verification

d. white (clear) box e. validation

11. Testing based on code coverage.

a black box b regression c. verification

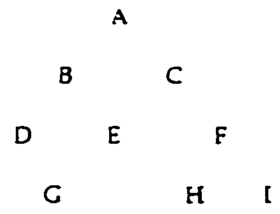
d. white (clear) box e. validation

12. Testing the interfaces among program parts.

a. black box b. integration c. verification

d. white (clear) box e. validation

The following questions deal with this tree:



13. Which traversal generates the order: D G B E A C H F I

a. preorder b. postorder c. inorder d. by level e. none

14. Which traversal generates the order: A B G D E C F H I

a. preorder b. postorder c. inorder d. by level e. none

15. Which traversal generates the order: A B D G E C F H I

a. preorder b. postorder c. inorder d. by level e. none

16. If this tree is a binary search tree ordered on a value not shown and the node A is to be deleted, where are the possible replacements for A?

a. node G and node I b. node E and node H

c. node E and node C d. node G and node H

e. node D and node I

*17. If N is the number of data elements to be manipulated by one or more algorithms, then the execution time of each such algorithm can be characterized in terms of N using the Big-O notation. Of the following, which best characterizes an execution time that is significantly different from the others?

a. O(N) b. O(N*N) c. O(N+2)

d. O(N/2) e. O(N+N)

18 The separation of the logical properties of data from the implementation details.

a. procedural abstraction b. data abstraction

c. data encapsulation d. information hiding

e. primogeniture

19 Six integer numbers are read. Each number is either printed or put on a stack. After the sixth number has been read and processed, the numbers on the stack are popped and printed. If the input values are 1, 2, 3, 4, 5, and 6 (in that order), which of the following lines of output is impossible?

a. 1 2 3 4 5 6 b. 1 3 5 4 6 2 c. 1 3 5 6 4 2

d. none e. all

20. Under the conditions described in question 19, which of the following lines of output is possible?

a. 6 5 4 3 1 2 b. 6 5 4 1 2 3 c. 1 2 4 6 5 3

d. none e. all

21. Six letters are read. Each letter is either printed or put on a FIFO queue. After the sixth letter has been read and processed, the letters on the queue are dequeued and printed. If the input letters are A, B, C, D, E, and F (in that order), which of the following lines of output is impossible?

a. A B C D E F b. A B D C F E c. A C E B D F

d. none e. all

22. Under the conditions described in question 21, which of the following lines of output is possible?

a. F E D C B A b. A B C F D E c. D E F C B A

d. none e. all

23. The ability of a program to recover following an error.

a. verification b. robustness c. abstraction

d. strength e. primogeniture

24. A special procedure or function that can stand in for a lower-level subprogram when testing.

a. driver b. recursive procedure c. stub

25. The order in which Pascal stores a two-dimensional arrays.

a. row major b. column major c. depends on the compiler

26. The EnQueue operation for the Queue ADT is an example of a(n)

a. iterator b. observer c. constructor

27. Which of the following variables is created at compile time?

a. global variable b. referenced variable

c. activation record d. variable on the run-time stack

e. local variable

28. Which of the following variables is created at run time?

a. local variable b. referenced variable

c. activation record d. variable on the run-time stack

e. all are created at run time

29. The sort whose invariant is:
Data[1]..Data[I-1] is sorted AND Data[I]..Data[N] is unexamined.

a. selection sort b. quick sort c. bubble sort

d. heap sort e. insertion sort

30. The sort whose invariant is:
Data[1]..Data[I-1] is sorted AND is <= Data[I]..Data[N].

a. merge sort b. quick sort c. bubble sort

d. insertion sort e. fibonacci sort

31. The best sort to use when the data is almost sorted.

a. merge sort b. quick sort c. bubble sort w/flag

d. heap sort e. insertion sort

*32. In QuickSort, if the splitting value is the first element in the array, then QuickSort will be most efficient with input data in:

a. sorted order

b. random order

c. nearly sorted order

d. reverse order

e. law and order

33. The NlogN sort that you should not use if stability is required.

a. merge sort b. quick sort c. bubble sort w/flag

d. heap sort e. insertion sort

*34. Merge sort (internal) has which of the following advantages over bubble sort for long lists?

i. Merge sort requires much less coding to implement than bubble sort

ii. Merge sort runs faster than bubble sort

iii. Merge sort requires less memory space than bubble sort

a. i only b. ii only c. iii only

d. i and iii e. ii and iii

35 If you want a listing by name within class, you sort first by

a. class b. name c. it doesn't matter

36 The order of inserting into a list implemented as an ordered linked list.

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. $O(N^2)$

37 The order of inserting into a list implemented in an ordered array.

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. $O(N^2)$

38. The order of merging two ordered lists (all implementations are the same).

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. $O(N^2)$

39. The order of inserting into a priority queue implemented as a heap.

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. $O(N^2)$

40. The order of creating an empty stack implemented as linked list.
a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. $O(N^2)$

41. The order of creating an empty priority queue implemented as a heap.

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. $O(N^2)$

42. In a procedure, value parameters do not necessarily protect the contents of the caller's data structures from being affected by execution of the procedure under which of the following conditions?

a. The procedure is recursive

b. The value parameters are integers

c. The value parameters are pointers

d. The value parameters are arrays

e. The procedure is used with a FORWARD declaration

Questions 43 - 48 deal with the following problem.

You have a situation where you need to use two stacks, S1 and S2. You know that together they will never have more than Max elements. You decide to use an array representation with both stacks residing in the same array but with two logical pointers to top, Top1 (for S1) and Top2 (for S2). Notice this will not be a linked list. Use the following definitions and declarations.

```
CONST
    Max = 100;
    MaxPlus1 = 101;
  TYPE
    ItemType = (* component on the stack *)
    StackType = RECORD
                  Stack : ARRAY[1..Max] OF ItemType;
                  Top1 : 0..Max;
                  Top2 : 1..MaxPlus1;
   END;   (* RECORD *)
```

*43. Choose the statement that correctly implements this procedure.

```
PROCEDURE ClearS2(VAR S2 : StackType);
BEGIN
  (* statement goes here *)
END;
```

a. S2.Top2 := 0

b. S1.Top1 := 0

c. S2.Top2 := MaxPlus1

d. S2.Top2 := Max

e. none is correct

*44. Choose the statement that correctly implements this procedure.

```
FUNCTION Full(S1, S2 : StackType) : BOOLEAN;
(* If S1 and S2 occupy the entire array, Full is TRUE *)
BEGIN
  (* statement goes here *)
END;
```

a. Full := S1.Top1 = Max

b. Full := S1.Top1 = S2.Top2

c. Full := S1.Top1 - S2.Top2 = 1

d. Full := S1.Top1 + S2.Top2 = MaxPlus1

e. none is correct

Questions 45 - 46 deal with the following procedure shell.

```
PROCEDURE PushS1(VAR S1 : StackType; Item : ItemType);
BEGIN
  S1.Top1 := __________________; (* question 45 *)
  ___________________________ := Item  (* question 46 *)
END;
```

*45 Choose the expression that makes the first statement correct.

a. S1.Top1 + 1

b. S1.Top1 - 1

c. Max - S1.Top1

d. S1.Top1 + MaxPlus1

e. none is correct

*46. Choose the expression that makes the second statement correct.

a. Stack[S1.Top1+1]

b. Stack[MaxPlus1 - S1.Top1]

c. Stack[S1.Top1]

d. S1.Stack[S1.Top1]

e. none is correct

Questions 47 - 48 deal with the following procedure shell.

```
PROCEDURE  PopS2(VAR S2 : StackType; VAR Item : ItemType);
 BEGIN
   Item := _____________________; (* question 47 *)
   S2.Top2 := ___________________  (* question 48 *)
 END;
```

*47. Choose the expression that makes the first statement correct.

a. Stack[S2.Top2]

b. Stack[Max - S2.Top2]

c. S2.Stack[Top2]

d. S2.Stack[S2.Top2]

e. none is correct

*48. Choose the expression that makes the second statement correct.

a. S2.Top2 + 1

b. S2.Top2 - 1

c. MaxPlus1 - Top2

d. Max - S2.Top2

e. none is correct

49 An array contains the following values: P R C Q W B D A T
The values are inserted (in the above order) into a priority queue implemented as a heap. Which is the resulting heap array?

a. W T D R Q B C A P

b. R W D T P B C A Q

c. W T D P R C B A Q

d. W T R Q C B D A P

e. none

Questions 50 - 53 deal with the binary search tree built from entering the following letters P R C Q W B D A T S U (in that order).

50. Which is the inorder successor of P?

a. Q b. W c. T d. P e. S

51. Which is the inorder predecessor of R?

a. R b. D c. A d. Q e. unknown

52. What is the height of the tree if the height of an empty tree is 0?

a. 2 b. 3 c. 4 d. 5 e. unknown

53. If W were deleted, what would be the right child of R?

a. T b. S c. U d. Q e. unknown

*54. Consider the following four tasks:

1. To perform a linear search of a list of N names
2. To perform a binary search of a sorted list of N names
3. To perform a selection sort into alphabetical order of a list of N names that are initially in random order
4. To perform a merge sort into alphabetical order of a list of N names that are initially in random order

For large N, which of the following lists these tasks in order (from least to greatest) or their worst-case running times?

a. 2, 4, 3, 1

b. 4, 3, 1, 2

c. 2, 4, 1, 3

d. 2, 1, 4, 3

e. 2, 3, 1, 4

Questions 55 - 57 use the following declarations

```
TYPE
  NodePtr = ^Node;
  Node = RECORD
            Info : INTEGER;
            Next : NodePtr
          END;
```

55 Suppose that the only Pascal implementation available does not support pointers or dynamic storage allocation using NEW. You are given a Pascal program that uses pointers and NEW for creating and manipulating linked structures. Your job is to modify the program so that it can be run using the available resources. What should you do?

a. simulate pointers by using trees and recursion

b. simulate pointers by using a combination of value parameters and VAR parameters

c. modify the Pascal implementation so that it does support pointers and NEW

d. simulate pointers by using indices into a large array and change the program accordingly

e. identify the queues and stacks in the program, and reimplement each using an array

56. List is an external pointer to a linked list implemented without a header node. If the list is empty, which of the following conditions is both meaningful and true?

```
a.  List = NIL
b.  List^.Next = NIL
c.  List^.Next = List
d.  List = 0
e.  Next = NIL
```

57. A method of protecting data within an ADT by only allowing access via the operations defined in the ADT.

a. procedural abstraction b. data abstraction

c. data encapsulation d. information hiding

e. primogeniture

*58. function z(k, n : integer) : integer;

```
begin
  if n = k
     then  z := k
  else
    if n > k
       then  z := z(k, n-k)
       else  z := z(k-n, n)
end;
```

Based on the function defined above, what is the value of z(6,8)?

a. 1 b. 2 c. 3

d. 4 e. 8

*59 It would be most appropriate to use a recursive function or procedure to solve a problem that

a. can be reduced to two or more simpler or smaller cases of the same problem

b. involves a substantial number of conditionals and nested loops

c. requires a lot of memory

d. involves storing data in a two-dimensional array

e. involves evaluation of the factorial function

*60.
```
 program Main;
 var z:integer;
 function F(x:integer);
 begin
   if (x = 1) or (x = 3)
      then  F := x
      else  F := x * F(x - 1)
 end;
begin  (* main *)
  z := F(F(2) + F(5))
end.
```

If MaxInt were large enough to allow the program above to be executed, then at the end of the program, the value of z would be

a. 62

b. 5! + 2!

c. (5! + 2!)!

d. (7!)!

e. (62!)/(2!)

*61. Of the following data representations for storing integers with an arbitrary number of digits, which would allow two integers having that representation to be added and the result to be stored using the same representation in the most time-efficient manner?

a. A binary search tree with the nodes containing the digits and their positions in the decimal representation of the integer and with the nodes ordered by the digits

b. A stack of digits, with the leftmost (most significant) digit on top of the stack

c. A stack of digits, with the rightmost (least significant) digit on the top of the stack

d. A linked list of digits, proceeding from the leftmost (most significant) digit to the rightmost (least significant) digit

e. A linked list of digits, proceeding from the rightmost (least significant) digit to the leftmost (most significant) digit

Questions 62 - 66 use the following array of integers.
20 9 10 3 6 21 8

62. The state of the array after Data[1]..Data[3] is sorted:
9 10 20 3 6 21 8.
Which sort is used?

a. straight selection b. bubble c. heap

d. insertion e. none of these

63. The state of the array after Data[1]..Data[2] is sorted:
3 6 10 20 9 21 8
Which sort is used?

a. straight selection b. bubble c. heap

d. insertion e. none of these

64. The state of the array after Data[1]..Data[2] is sorted:
3 6 20 9 10 8 21
Which sort is used?

a. straight selection b. bubble c. heap

d. insertion e. none of these

65. Quick sort is being used with Data[1] as the split value. What is the state of the array when the first recursive call is made?

a. 6 9 10 3 20 8 21 b. 8 9 10 3 6 20 21

c. 20 9 10 3 6 21 8 d. unknown

66. What is the state of the run-time stack when the first recursive call is made? The base address of the array is 100.

	Data	First	Last	Return
a.	100	1	7	R1
	100	1	6	R2 <-- Top of Stack
b.	100	100	106	R1
	100	100	104	R2 <-- Top of Stack
c.	100	1	5	R2
	100	1	7	R1 <-- Top of Stack
d.	100	1	7	R1

100 1 5 R2 <-- Top of Stack

e. no recursive call is made

Questions 67 - 68 deal with the following problem. The height of a binary tree is the number of nodes in the longest path from the root to a leaf of the tree. The height of an empty tree is 0; the height of a single-node tree is 1

```
1  Function Height(Tree : TreeType) : INTEGER;
2  Begin
3    If __________________        (* question 67 *)
4      then
5        Height := 0
6      else
7        _________________________ (* question 68 *)
8  end;
```

*67 Fill the blank in line 3.

a. Tree <> NIL b. Tree = NIL

c. Tree^ = NIL d. Tree^.Next = NIL

e. none of the above

*68. Fill the blank in line 7.

a. Height(Tree^.Right) + Height(Tree^.Left)

b. Height(Tree^.Right) + Height(Tree^.Left) + 1

c. Height := Height(Tree^.Right) + Height(Tree^.Left) +1

d. Height := Max(Height(Tree^.Right),Height(Tree^.Left)) +1

e. none of the above

69. The language feature that is necessary for recursion.

a. dynamic variables b. run-time stack c. procedures

d. NEW/DISPOSE e. robustness

Questions 70-74 use the following declarations. You may assume that all data types take one memory location.

```
CONST
  Limit = 100;

TYPE
  ColorType = (blue, white, silver, red, black);
  NameString = String[15];
  CarRecord = RECORD
                 Model : NameString;
                 Color : ColorType;
                 Cost : REAL;
                 Doors: (two, four);
                 IDNum: INTEGER;
                 Sold : BOOLEAN
               END;
  Inventory = ARRAY [1..Limit] OF CarRecord;

VAR
  Car : CarRecord;
  InStock : Inventory;
  Index : INTEGER;
```

70. **Which of the following expressions accesses the hundreds position of the IDNum of Car?**

    a. Car.IDNum[3]

    b. (Car.IDNum MOD 100)

    c. (Car.IDNum DIV 100)

    d. ((Car.IDNum MOD 100) DIV 10)

    e. ((Car.IDNum DIV 100) MOD 10)

71. **If Car begins at location 1, what is the location of Car.Model[3]?**

    a. 0 b. 1 c. 2 d. 3 e. unknown

72. **If Car begins at location 1, what is the location of Car.IDNum?**

    a. 3 b. 18 c. 19 d. 20 e. unknown

73. **If Car begins at location 1, what is the location of InStock[2].Model[2]?**

    a. 24 b. 41 c. 42 d. 27 e. unknown

74. **If Car begins at location 1, what is the location of Index?**

    a. 2000 b. 2020 c. 2021 d. 2022 d. unknown

Question 75-77 use the following table.

N	column a	column b	column c
32	5	1024	160
64	6	4096	384
128	7	16384	896
256	8	65536	2048
512	9	262144	4608
1024	10	1048576	10240
2048	11	4194304	22528
4096	12	16777216	49152

You have run different algorithms to accomplish the same task and these were the values that were returned as an amount of work for the appropriate value of N (first column).

•75. Which of the following would most closely describe the complexity of the algorithm returning the values in column a?

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. O(N^2)

•76. Which of the following would most closely describe the complexity of the algorithm returning the values in column b?

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. O(N^2)

•77. Which of the following would most closely describe the complexity of the algorithm returning the values in column c?

a. O(1) b. O(logN) c. O(N) d. O(NlogN) e. O(N^2)

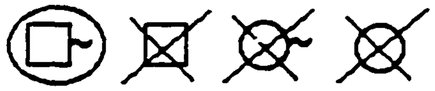
Appendix E

Propositional Learning Test (PLT) with Instructions

Propositional Logic Test (PLT)
Instructions

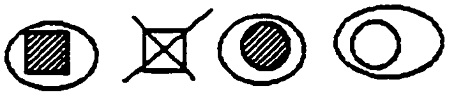
In each of the problems on the PLT you will find a sentence followed by four figures. Each figure is either square or round, either large or small, either white or striped, and either tailed (has a tail) or untailed. Your task is to circle those figures that are allowed by the sentence and to cross out the ones that are not allowed. Here are some examples with the correct answers to show what this means. Study them carefully since the problems that follow are very similar.

Ex 1. It is square and it is tailed.



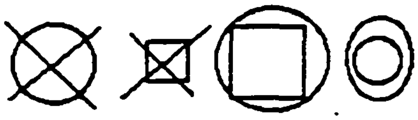
Here it says it must be square and tailed so only the one that is both square and tailed fits. The others are not tailed or are not square or are not both so that they should be crossed out.

Ex 2. If it is white then it is round.



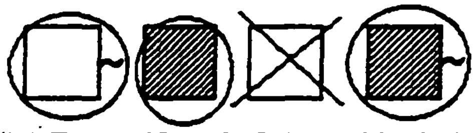
If it is White then it must be round, but if it is striped then it doesn't matter if it's round or not. So the white circle fits but the white square does not. The striped figures all fit because the statement only tells us about white figures.

Ex 3. If it is round it is small and if it is small it is round.



The round ones that are small fit and so do the small ones that are round. Since the large square isn't round it doesn't have to be small, so it fits. The large circle doesn't fit the first part of the rule and the small square doesn't fit the second part.

Ex 4. It is striped or it is tailed or both.

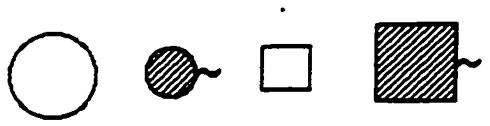


You can circle the first figure because it is tailed. The second figure also fits because it is striped, and the last one fits because it is both striped and tailed. The third figure doesn't fit since it is neither striped nor tailed.

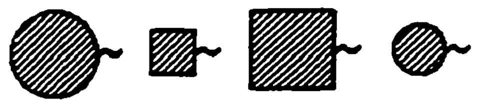
You have 15 minutes to complete the test.

Name:____

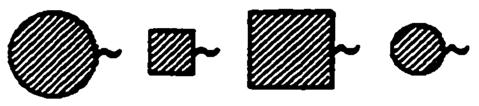
1. It is round and it is striped.



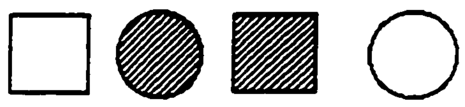
2. It is small or it is round or both.



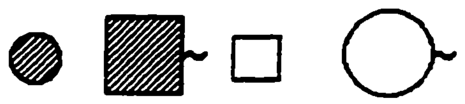
3. If it is large then it is round.



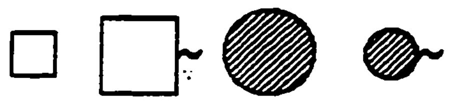
4. If it is round it is striped and
if it is striped it is round.



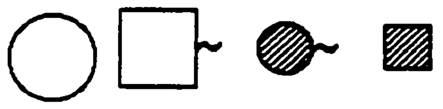
5. It is tailed or it is square or both.



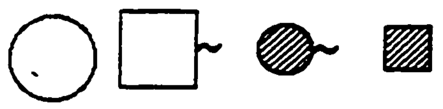
6. It is striped and it is large.



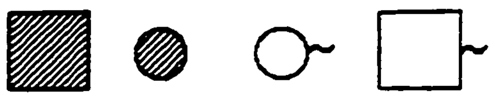
7. If it is large it is square and
if it is square it is large.



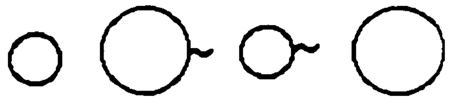
8. If it is small then it is square.



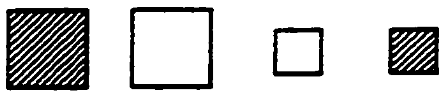
9. It is large and it is tailed.



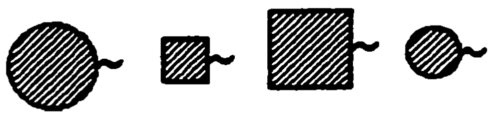
10. It is large or it is untailed or both.



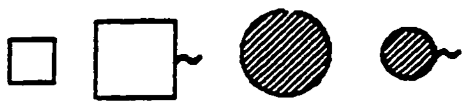
11. If it is white then it is large.



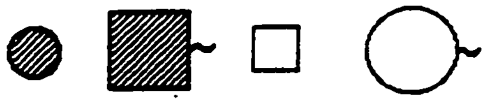
12. If it is small it is square and
if it is square it is small.



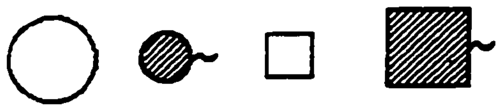
13. It is striped or it is small or both.



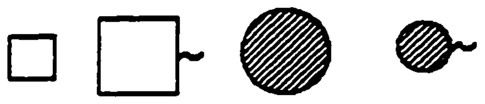
14. It is tailed and it is round.



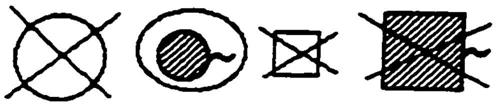
15. If it is square it is white and
if it is white it is square.



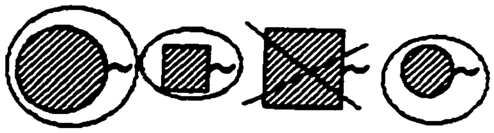
16. If it is striped then it is large.



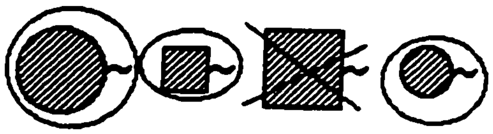
1. It is round and it is striped.



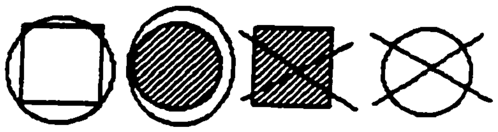
2. It is small or it is round or both.



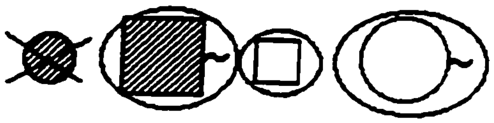
3. If it is large then it is round.



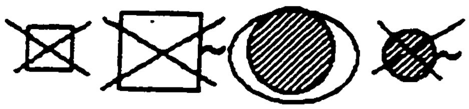
4. If it is round it is striped and
if it is striped it is round.



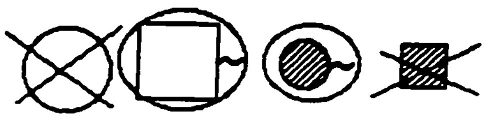
5. It is tailed or it is square or both.



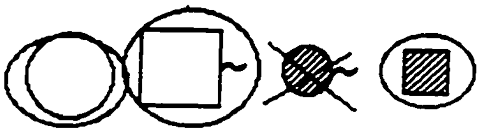
6. It is striped and it is large.



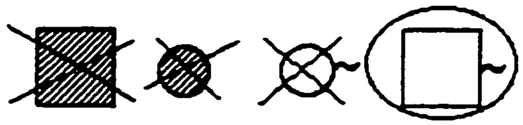
7. If it is large it is square and
if it is square it is large.



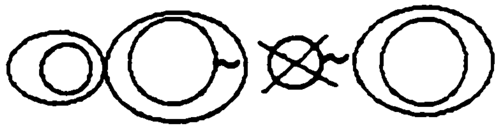
8. If it is small then it is square.



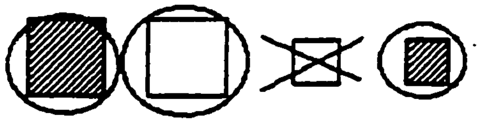
9. It is large and it is tailed.



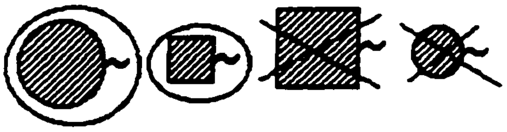
10. It is large or it is untailed or both.



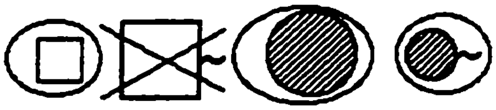
11. If it is white then it is large.



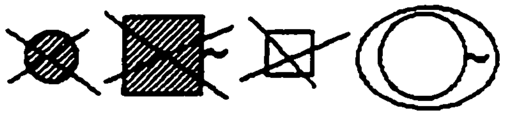
12. If it is small it is square and
if it is square it is small.



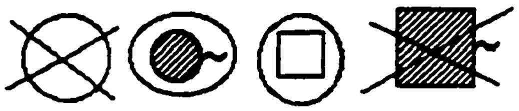
13. It is striped or it is small or both.



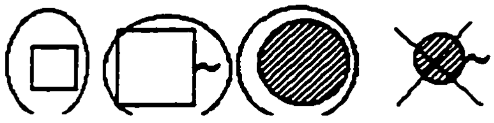
14. It is tailed and it is round.



15. If it is square it is white and
if it is white it is square.



16. If it is striped then it is large.



Scoring and Interpretation of the PLT

Scoring of a completed form: Using the key, each of the 16 items should be checked for correctness using either the key or the grayed-in row of the appropriate table on pages 3 and 4. A pen of a contrasting color can be used to show the correct response directly on the completed form. A "number-correct" indicator such as "X/16" can be given at the top of the page, although this number may be misleading to the test-taker since the *pattern* of incorrect responses is actually much more interesting than the count of correct/incorrect.

Interpretation of results: The table at the top of page 2 provides details about the system of 16 binary operations upon which the PLT is based. The first column, marked "Piaget's Notation", comes directly from the PLT Key (1989). The information in the second column gives names to the 16 different operations based on different sources: PLT is the operation name as defined in the key for the Propositional Logic Test, 1989; G&O is the operation name as defined by Ginsburg & Opper, 1979, p. 191.

To interpret the results, refer to the charts on pages 3 and 4 of this packet. There is a table for each of the four subtests, with the item breakdown as follows:

- conjunction: items 1, 6, 9, & 14
- disjunction: items 2, 5, 10, & 13
- implication: items 3, 8, 11, & 16
- biconditional: items 4, 7, 12, & 15

For each incorrect response, refer to the proper column of the proper table and find the pattern that matches the *actual* response; the name of the operation that the response indicates can be written next to the problem.

Once all of the incorrectly answered items have been considered, you will be able to judge whether errors were systematic or not. Non-systematic errors could be due to carelessness, inattention, or mis-reading of the statement. The diagnosis of systematic errors can be especially helpful for the instructor in designing lesson plans or providing remediation.

Example: Suppose that on item #3, the respondent gave the answer OXXX rather than OOXO. This would tell you that the respondent treated this statement as a conjunction rather than as an implication; that is, this person answered the item "If it is large then it is round" as if it said "It is large and it is round". By considering all of the items on the implication subtest (items 3, 8, 11, & 16), the instructor/researcher can detect whether the respondent has misinterpreted the items having to do with implication.

Related literature:

Ginsberg, H., & Opper, S. (1979). *Piaget's theory of intellectual development* (2nd ed.). Englewood Cliffs, NJ: Prentice-Hall.

Piburn, M. D. (1989). Reliability and validity of the Propositional Logic Test. *Educational and Psychological Measurement, 49*, 667–672.

Piburn, M. D. (1990). Reasoning about logical propositions and success in science. *Journal of Research in Science Teaching, 27*(9), 887-900.

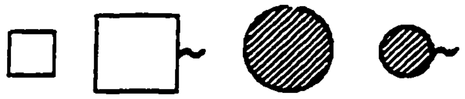
PLT Key. (1989). Key to the Propositional Logic Test. Unpublished document. Obtained through M. Piburn, Arizona State University.

Appendix F

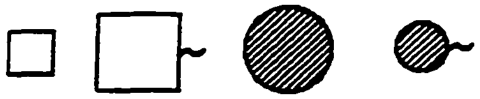
Alternate Propositional Learning Test (A-PLT)

Name:______________________________________

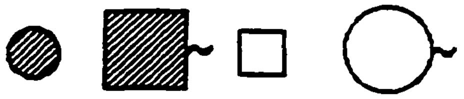
1. It is striped or it is small or both.



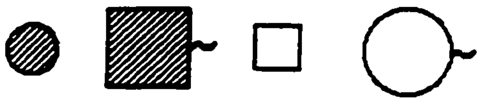
2. If it is striped then it is large.



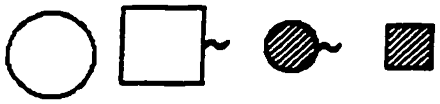
3. It is tailed or it is square or both.



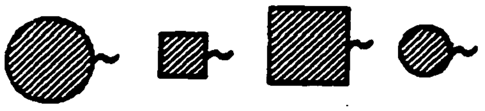
4. It is tailed and it is round.



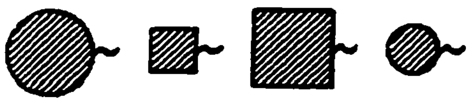
5. If it is small then it is square.



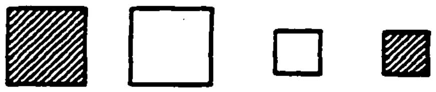
6. If it is small it is square and
   if it is square it is small.



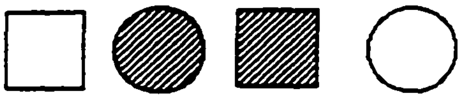
7. If it is large then it is round.



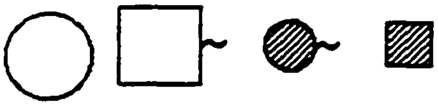
8. If it is white then it is large.



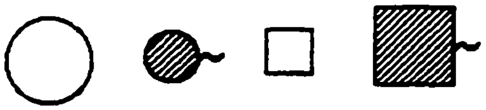
9. If it is round it is striped and
   if it is striped it is round.



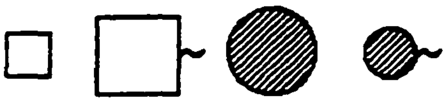
10. If it is large it is square and
    if it is square it is large.



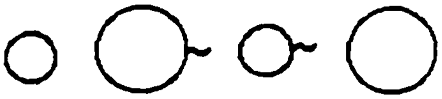
11. It is round and it is striped.



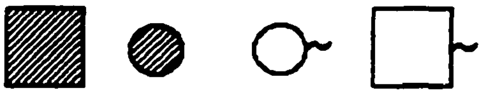
12. It is striped and it is large.



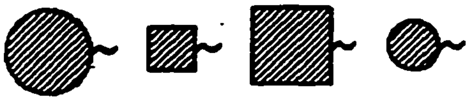
13. It is large or it is untailed or both.



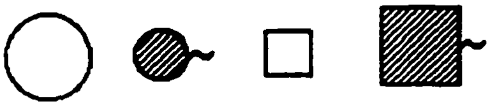
14. It is large and it is tailed.



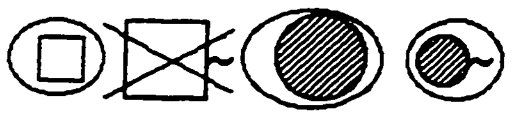
15. It is small or it is round or both.



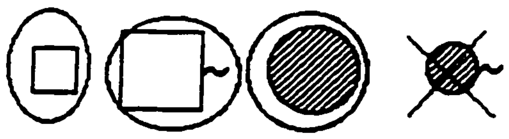
16. If it is square it is white and
    if it is white it is square.



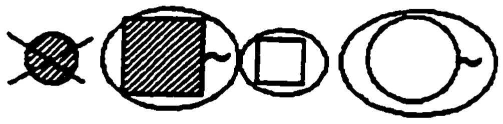
1. It is striped or it is small or both.



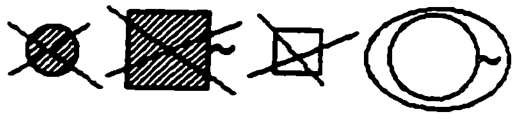
2. If it is striped then it is large.



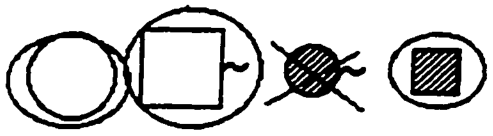
3. It is tailed or it is square or both.



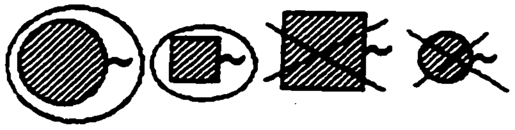
4. It is tailed and it is round.



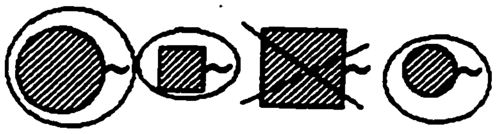
5. If it is small then it is square.



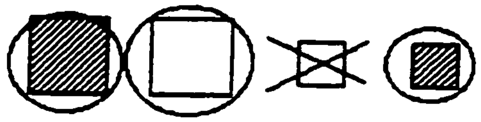
6. If it is small it is square and
if it is square it is small.



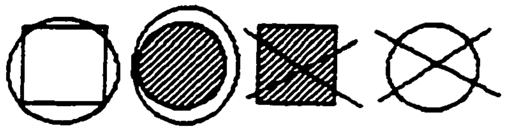
7. If it is large then it is round.



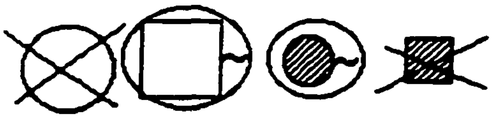
8. If it is white then it is large.



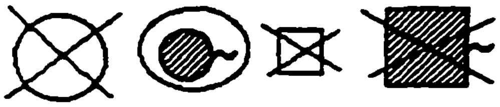
9. If it is round it is striped and
if it is striped it is round.



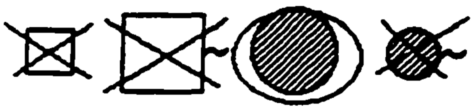
10. If it is large it is square and
if it is square it is large.



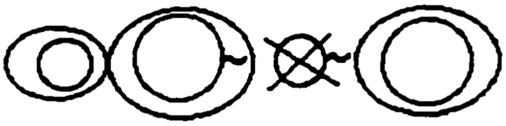
11. It is round and it is striped.



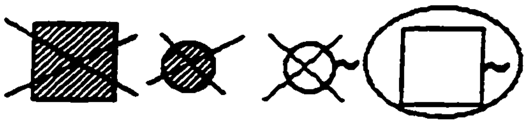
12. It is striped and it is large.



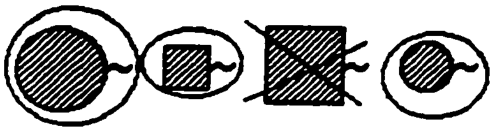
13. It is large or it is untailed or both.



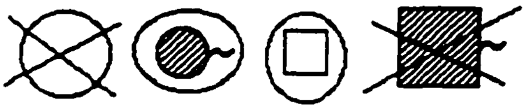
14. It is large and it is tailed.



15. It is small or it is round or both.



16. If it is square it is white and
if it is white it is square.



Appendix G

CS 315 Group Evaluation Form

CS 315
Group Evaluation Form
Confidential

Name _______________

Who were the members of your working group?

List at least one positive aspect of your group's performance?

How could the group have performed better?

Please estimate the percentage of effort each of your group members (including yourself) contributed to the programs and homework. (The total should add up to %100).

You may make any additional comments on the back.

Appendix H

External Cooperative Learning Review

CS 315 Cooperative Learning Review

Reviewer ____________________

Date ________________

Does the learning environment you observed correspond to your definition of a cooperative learning environment?

Lowell J. Bethel, 10:01 PM 8/5/96 -, Re: Cooperative Learning stuff

Date: Mon, 5 Aug 1996 22:01:51 -0800
To: Roger Priebe <priebe@cs.utexas.edu>
From: lbethel@mail.utexas.edu (Lowell J. Bethel)
Subject: Re: Cooperative Learning stuff

Below is the recorded observation that I made of your class last month. I hope that it is acceptable. Have a safe and good trip back home/Bethel

```
>I observed Roger Priebe's computer science class on July 10, 1996. It was
>held in the Sanchez education building in the afternoon. As I entered the
>room I observed students divided into groups of about three students each.
>The groups appeared to be focused and engaged in conversation around
>different problems. Mr. Priebe was observed moving around the classroom
>periodically answering questions that were raised by students in the
>groups. During my observation, Mr. Priebe stopped the class and conducted
>a short lecture and discussion around one or two problems that the groups
>appeared to be discussing in their own groups. After about three-five
>minutes the students returned to their individual groups and continued
>with their small groups discussions. I moved around the class with few if
>ant students paying much attention to my presence. I observed the groups
>working for about 45 minutes before I left to attend a meeting.
>
>It appeared to this observer that the students were working very
>cooperatively in small groups around computer science problems.
```

Lowell J. Bethel
Science Education Center, SZB 340D
The University of Texas at Austin
Austin, TX 78712
(512) 471-7354
Fax (512) 471-8466
E-mail: lbethel@mail.utexas.edu

James P. Barufaldi, 03:54 PM 7/31/96 , Re: Cooperative Learning Obser

Date: Wed, 31 Jul 1996 15:54:13 -0500
To: Roger Louis Priebe <priebe@cs.utexas.edu>
From: jamesb@mail.utexas.edu (James P. Barufaldi)
Subject: Re: Cooperative Learning Observation

Hello RP:

Thanks for inviting me to your class.

I believe that your class did reflect characteristics of a cooperative learning environment. I observed that groups of students were clustered around tables. Each group indicted a sharing type of interaction with much discussion. I observed groups of students talking with each other about tasks and assignments; some shared writings recorded in their notebooks; others shared print matter with their partners from various resources. The groups appeared positive and entusiastic toward their work and quite comfortable throughout their participation. You did establish a learning environment in which cooperation and collaborative learning are encouraged and valued. jB

****
Happy Sciencing!

jamesb@mail.utexas.edu
James P. Barufaldi, Director
Science Education Center SZB 340
The University of Texas at Austin
Austin, Texas 78712
512-471-7354
FAX: 512-471-8466

CS 315 Cooperative Learning Review

Reviewer: Henry M. Walker

Date: July 10, 1996

Does the learning environment you observed correspond to your definition of a cooperative learning environment?

From my perspective, *cooperative learning* is a process during which students interact with each other as part of their expected course-related activity. Similarly, a *cooperative learning environment* is any course framework in which cooperative learning is encouraged. Following this definition, cooperative learning may range from directed activities, where students work in a group following a highly-prescribed series of instructions, to undirected experiences, where students are given no instructions or guidance of any type. Thus, as two extremes, cooperative learning includes both group assignments involving highly structured exercises and completely unstructured study halls, in which students are free to interact in any way they wish on any topic they find of mutual interest.

Prior to visiting both sections of CS 345 on Tuesday, I imagined cooperative learning as involving a reasonably high level of guidance for students, with rather detailed instructions and agendas. I had not thought about an unstructured setting as serving as a cooperative learning environment. Thus, the classroom environment for yesterday's afternoon class was not at all what I had expected, for the class seemed to be some combination of study hall and informal office hours. On the other hand, after a (sometimes) long initial period, most class members did begin talking with each other about topics that often were related to the class. With the occurrence of this interaction among students, certainly the class I observed must fall within the general category of a cooperative learning environment. (To record two types of interactions, one group spent some time initially talking about a recent calculus test before addressing questions on an upcoming homework assignment for this course. Another group discussed an assignment for this class briefly and then spent most of the class discussing courses and requirements for the computer science major.) Thus, my observations have led me to start rethinking my definitions and perceptions of what cooperative learning is about. This experience also has started to expand my appreciation of what activities might be included within a general framework of cooperative learning.

Also, it may be worth noting that, in the past, I have considered cooperative learning as a mechanism to support a variety of learning styles, and I have read the reports of Triesman and others which indicate the particular value of this approach for various "at-risk" constituencies (e.g., women, African-Americans, Hispanics). In this regard, I noted that, in this experiment, most of the at-risk students seemed to be in the more traditional section of CS 315. For example, all African-Americans attending on Tuesday seemed to be in the traditional section. Also, in the cooperative learning section, it seems that only 6 of the 30+ students are women, all five who attended on Tuesday were Asian-American, and none talked to each other (although two were seated at the far ends of one group – separated by a male; the others were in different groups).

As an extended analogy, a *laboratory experience* is any context in which students are expected to perform some activities within a computer lab, using computer equipment in some way. Within this context, the term, *closed lab*, has come to mean a regularly scheduled activity, staffed by faculty or teaching assistants, and involving a prearranged sequence of activities to be carried out by students – often requiring the completion of written exercises.

Such closed labs need not focus upon the acquisition of skills (e.g., learning to use a debugger or editor), but rather on the principles of the scientific method, including experimentation, observation, data collection, data analysis, and hypothesis confirmation or rejection. In contrast, *open labs* involve students working on projects or assignments without specific, detailed instruction and without the formal staffing by faculty or other professionals. In this context, closed labs involve detailed instructions and the opportunity to consult experts, while open labs require only an open room containing machines available for students.

In the past, I have considered open labs as supporting individuals in their work on programming assignments, and my experience has provided some insights concerning what types of lab environments may be more helpful than others for this type of laboratory experience. Also, my past work teaching both calculus and computer science has suggested that the effectiveness of closed labs is heightened dramatically, particularly in beginning courses, if students are given extremely detailed directions. For example, beginning students have a difficult time with general instructions, such as "investigate the differences between value and reference parameters in the following program." Rather, the labs are much more productive if students are given detailed steps which can help them in the investigation. (With this guided experimentation in early courses, more experienced students at upper levels have a much better sense of how to plan appropriate investigations.) To be still more specific, my experience consistently shows that students can cover 10% to 20% more material (with correspondingly better test scores) at the beginning level with highly-prescribed, closed labs than with more open-ended, closed labs.

In describing this experience with labs, I do not mean to suggest that open labs are not ever appropriate – on the contrary, I think open labs are essential for various homework activities. However, my consistent experience has been that in-class lab time produces far better results if it is highly structured.

With this experience with labs, I had always assumed that in-class cooperative learning activities also would be highly structured; it simply never occurred to me that other types of in-class cooperative learning environments might also work very well. From this perspective, this experiment with two types of environments for CS 315 has the potential to provide extremely interesting results for comparing a reasonably-traditional class format with an open-ended, non-directed, cooperative learning environment. It is not so clear, however, to what extent this experiment can shed light on the effectiveness of the more structured cooperative learning environment which I had previously imagined.

Appendix I

Attendance

CS 315 Attendance

Date	Morning	Morning Possible	Pct.	Afternoon	Afternoon Possible	Pct.
June 10	32	32	1.00	33	33	1.00
June 11	31	31	1.00	33	33	1.00
June 12	29	31	0.94	30	33	0.91
June 17	27	31	0.87	30	33	0.91
June 18	31	31	1.00	33	33	1.00
June 19	29	31	0.94	30	33	0.91
June 24	28	31	0.90	31	33	0.94
June 25	27	31	0.87	30	33	0.91
June 26	28	31	0.90	30	32	0.94
July 1	22	31	0.71	31	32	0.97
July 2	27	31	0.87	30	32	0.94
July 3	23	31	0.74	26	32	0.81
July 8	21	31	0.68	30	32	0.94
July 9	27	30	0.90	27	31	0.87
July 10	20	30	0.67	29	31	0.94
July 15	24	30	0.80	24	31	0.77
July 16	25	30	0.83	26	31	0.84
July 17	27	30	0.90	26	31	0.84
July 22	25	30	0.83	27	31	0.87
July 23	20	29	0.69	25	31	0.81
July 24	20	29	0.69	23	30	0.77
July 29	21	29	0.72	25	30	0.83
July 30	23	29	0.79	28	30	0.93
	25.52	30.43	0.84	28.57	31.78	*0.90

*Significant at alpha=0.05 level

Bibliography

Almstrum, V. L. (1991). The relationship between pre-college mathematics and the undergraduate computer science curricula. SIGSCE Bulletin, 23(1), 124-129.

Almstrum, V. L. (1994). Limitations in the understanding of mathematical logic by novice computer science students. Unpublished doctoral dissertation, The University of Texas at Austin.

Aronson, E., Blaney, N., Stephan, C., Sikes, J., & Snapp, M. (1978). The jigsaw classroom. Beverly Hills, CA: Sage.

Ausubel, D. (1963). Psychology of meaningful verbal learning. New York: Grune & Straton.

Barker, R. J., & Unger, E. A. (1983). A prediction for success in an introductory programming class based upon abstract reasoning development. SIGCSE Bulletin, 15, 154-158.

Bligh, D. (1972). What's the use of lectures? Harmondsworth, England: Penguin.

Bloom, B. S. (Ed.). (1956). Taxonomy of educational objectives: The classification of educational goals. New York: David McKay Company, Inc.

Blumenfeld, P. C., Marx, R. W., Soloway, E., & Krajcik, J. (1996) Learning with peers: From small group cooperation to collaborative communities. Educational Researcher, 25 (8), 37-40.

Bossert, S. T. (1988-1989). Cooperative activities in the classroom. In E. Z. Rothkopf (Ed.), Review of research in education: Vol. 15 (pp. 225-252). Washington, DC: American Educational Research Association.

Brainerd, C. (1978). Piaget's theory of intelligence. Englewood Cliffs, NJ: Prentice Hall.

Bruner, J. (1960). The process of education. Cambridge: Harvard University Press.

Bruner, J. (1962). Introduction. In E. Hanfman & G. Vukar (Eds.). Thought and language. Cambridge, MA: MIT Press.

Burton, D. (1992). The effect of closed laboratory activities on the comprehension of five concepts and the perception of effectiveness of the course in a second semester computer science course. Unpublished doctoral dissertation, The University of Texas at Austin.

Campbell, D. T., & Stanley, J. C. (1963). Experimental and quasi-experimental designs for research. Chicago: Rand McNally.

Dale, N., & Lilly, S. C. (1995). Pascal plus data structures, algorithms, and advanced programming (4th ed.). Lexington, MA: D. C. Heath and Company.

DeVries, D. L., & Slavin, R. E. (1978). Teams-games-tournaments (TGT): Review of ten classroom experiments. Journal of Research and Development in Education, 12, 28-38.

Enyeart, M. A. (1980). Relationships among propositional logic, analytical reasoning, and Piagetian level. Unpublished doctoral dissertation, Rutgers University.

Gabbert, B., Johnson, D. W., & Johnson, R. (1986). Cooperative learning, group-to-individual transfer, process gain, and the acquisition of cognitive reasoning strategies. Journal of Psychology, 120(3), 265-278.

Gallagher, J. M., & Reid, D. K. (1981). The learning theory of Piaget and Inhelder. Monterey, CA: Brooks/Cole.

Grisham, D. L., & Molinelli, P. M. (1995). Cooperative learning. Westminster, CA: Teacher Created Materials, Inc.

Guzdial, M. (1995). Centralized mindset: A student problem with object-oriented programming. ACM SIGCSE Bulletin, 27(1), 182-185.

Hudak, M. A., & Anderson, D. E. (1990). Formal operations and learning styles predict success in statistics and computer science courses. Teaching of Psychology, 17(4), 231-234.

Jaramillo, J. A. (1996). Vygotsky's sociocultural theory and contributions to the development of constructivist curricula. Education, 117 (1), 133-140.

Johnson, D. W., & Johnson, R. T. (1985). The internal dynamics of cooperative groups. In R. Slavin, S. Sharan, S. Kagan, R. H. Lazarowitz, C. Webb, & R. Schmuck (Eds.), Learning to cooperate, cooperating to learn. New York: Plenum Press.

Johnson, D. W., & Johnson, R. T. (1989). Cooperation and competition: Theory and research. Edina, MN: Interaction Book Company.

Johnson, D. W., & Johnson, R. T. (1994). Learning together and alone: Cooperative, competitive, and individualistic learning (4th ed.). Edina, MN: Interaction Book Company.

Johnson, D. W., Johnson, R. T., & Smith, K. A. (1991). Active learning: Cooperation in the college classroom. Edina, MN: Interaction Book Company.

Kim, Y. (1995). The reasoning ability and achievement of college level students enrolled in a logic class in computer science. Unpublished doctoral dissertation, The University of Texas at Austin.

Klein, J. D., & Pridemore, D. R. (1992). Effects of cooperative learning and need for affiliation on performance, time on task, and satisfaction. Educational Technology Research and Development, 40(4), 39-47.

Kleiner, K. A. (1992). Collaborative learning is possible in larger courses. In S. J. Hamilton & E. Hanson (Eds.), Collaborative learning: Sourcebook for collaborative learning in the Arts and Sciences at Indiana University. ERIC Accession Number ED347914.

Kulik, J., & Kulik, C. L. (1979). College teaching. In P. L. Peterson & H. J. Walberg (Eds.), Research on teaching: Concepts, findings, and implications. Berkeley, CA: McCutcheon.

Kurtz, B. I. (1980). Investigating the relationship between the development of abstract reasoning and performance in an introductory programming class. SIGCSE Bulletin, 12, 110-117.

Land, S. M., & Hannafin, M. J. (1996). Student-centered learning environments: Foundations, assumptions, and implications. ERIC Accession Number ED397810.

Lazar, A. M. (1995). Who is studying in groups and why? Peer collaboration outside the classroom. College Teaching, 43(2), 61-65.

Lew, M., Mesch, D., Johnson, D. W., & Johnson, R. (1986a). Positive interdependence, academic and collaborative-skills group contingencies and isolated students. American Educational Research Journal, 23, 476-488.

Lew, M., Mesch, D., Johnson, D. W., & Johnson, R. (1986b). Components of cooperative learning: Effects of collaborative skills and group contingencies on achievement and mainstreaming. Contemporary Educational Psychology, 11, 229-239.

Mason, D. (1997, March). Gateway to success for at-risk students in a large-group introductory chemistry class. Presented at the Annual Meeting of the National Association for Research in Science Teaching, Oak Brook, IL.

McKeachie, W. (1988). Teaching thinking. Update, 2(1), 1.

McKeachie, W., & Kulik, J. (1975). Effective college training. In F. Kerlinger (Ed.), Review of Research in Education. Itasca, IL: Peacock.

Mehta, J. I. (1993). Cooperative learning in computer programming at the college level. Unpublished doctoral dissertation, The University of Illinois, Chicago.

Nickerson, R. S., Perkins, D. N., & Smith, E. E. (1985). The teaching of thinking. Hillsdale, NJ: Erlbaum.

Nist, S. L., Holschuh, J. L., & Sharman, S. J. (1995). Making the grade in undergraduate biology courses: factors that distinguish high and low achievers. ERIC Accession Number ED390934.

Park, I. (1993). Cooperative learning and individual learning with computer assisted instruction in an introductory university level chemistry course. Unpublished doctoral dissertation, The University of Texas at Austin.

Penner, J. (1984). Why many college teachers cannot lecture. Springfield, IL: Charles C. Thomas.

Petress, K. C. (1996). The dilemma of university undergraduate attendance policies: To require class attendance or not. College Student Journal, 30, 387-389.

Piaget, J. (1962). Comments on Vygotsky's critical remarks concerning The language and thought of the child and Judgement and reasoning in the child. In L. S. Vygotsky, Thought and language. Cambridge: MIT Press.

Piburn, M. D. (1985). A test of propositional reasoning ability. In Educational Research: Then and Now. Hobart, Tasmania: Australian Association for Research in Education.

Piburn, M. D. (1989). Reliability and validity of the propositional logic test. Educational and Psychological Measurement, 49, 667-672.

Piburn, M. D. (1990). Reasoning about logical propositions and success in science. Journal of Research in Science Teaching, 27(9), 887-900.

Purdom, D. M., & Kromrey, J. D. (1995). Adapting cooperative learning strategies to fit college students. College Student Journal, 29, March 1995.

Roth, W. M. (1990). Collaboration and constructivism in the classroom. Paper presented at the Annual Convention of the American Educational Research Association, Boston, MA.

Ruggerio, V. R. (1988). Teaching thinking across the curriculum. New York: Harper & Row.

Sabin, R., & Sabin, E. (1994). Collaborative learning in an introductory computer science course. The Papers of the 25th SIGCSE Technical Symposium on Computer Science Education, 304-308.

Sharan, S. (1990). Cooperative learning: a perspective on research and practice. In S. Sharan, (Ed.) Cooperative learning: theory and research. New York: Praeger.

Sharan, S., & Sharan, Y. (1976). Small-group teaching. Englewood Cliffs, NJ: Educational Technology Publications.

Skon, L., Johnson, D. W., & Johnson, R. (1981). Cooperative peer interaction versus individual competition and individualistic efforts: Effects on the acquisition of cognitive reasoning strategies. Journal of Educational Psychology, 73(1), 83-92.

Slavin, R. (1978). Student teams and achievement divisions. Journal of Research and Development in Education, 12, 39-49.

Slavin, R. (1985). An introduction to cooperative learning research. In R. Slavin, S. Sharan, S. Kagan, R. H. Lazarowitz, C. Webb, & R. Schmuck (Eds.), Learning to cooperate, cooperating to learn. New York: Plenum Press.

Slavin, R. (1990). Cooperative learning: Theory, research, and practice. Englewood Cliffs, NJ: Prentice Hall.

Tenenberg, J. D. (1995). Using cooperative learning in the undergraduate computer science classroom. Journal of Computing in Small Colleges, 11(2), 38-49.

Tobin, K., Tippins, D. J., & Gallard, A. J. (1994). Research on instructional strategies for teaching science. In D. L. Gabel (Ed.), Handbook of research on science teaching and learning. New York: Macmillan.

Toothacker, W. S. (1983). A critical look at introductory laboratory instruction. American Journal of Physics, 51, 516-520.

Treisman, P. (1985). A study of the mathematics performance of Black students at the University of California, Berkeley. Unpublished doctoral dissertation, University of California, Berkeley.

Tucker, A. (1991). Computing curricula 1991. Report for the ACM/IEEE joint curriculum task force.

Van Blerkom, M. L. (1992). Class attendance in undergraduate courses. The Journal of Psychology, 126(5), 487-494.

Vygotsky, L. S. ([1934] 1962). Thought and language. E. Hanfman, & G. Vukar (Eds.). Cambridge: MIT Press.

Vygotsky, L. S. ([1930s] 1978). Mind in society. M. Cole, V. John-Steiner, S. Scribner, & E. Souberman (eds.). Cambridge, MA: Harvard University Press.

Wales, C., & Sager, R. (1978). The guided design approach. Englewood Cliffs, NJ: Educational Technology Publications.

Walker, H. M. (1997). Collaborative learning: A case study for CS 1 at Grinnell College and UT-Austin. ACM SIGCSE Bulletin, 29(1), 209-213.

Wertsch, J. V. (1991). Voices of the mind: A sociocultural approach to mediated action. Cambridge: Harvard University Press.

Whitman, N. A. (1988). Peer teaching: To teach is to learn twice. ASHE-ERIC Higher Education Report No. 4. Washington, DC: Association for the Study of Higher Education.

Yerion, K., & Rinehart, J. A. (1995). Guidelines for collaborative learning in computer science. ACM SIGCSE Bulletin, 27(4), 29-34.

VITA

Roger Louis Priebe was born in Tioga, North Dakota, on April 26, 1965, the son of Rodney and Marlene Priebe. He received Bachelor of Science (1987) and Master of Science (1990) degrees in Computer Science from the University of North Dakota, Grand Forks. During the following years he was employed by Stetson University, DeLand, Forida, as a Lecturer in the Department of Math/Computer Science. In August of 1994 he entered the Graduate School of The University of Texas at Austin in pursuit of a Ph.D. in Computer Science Education. During this time he was employed as an Assistant Instructor in the Department of Computer Science and as a contract employee at Dell Computers.

Permanent Address: Rt. 1, McGregor, ND 58755

This dissertation was typed by the author.

IMAGE EVALUATION TEST TARGET (QA-3)