THE EFFECTIVENESS OF USING LEGO® MINDSTORMS® ROBOTICS ACTIVITIES TO INFLUENCE SELF-REGULATED LEARNING IN A UNIVERSITY INTRODUCTORY COMPUTER PROGRAMMING COURSE

William Isaac McWhorter, B.S., M.S.

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

May 2008

APPROVED:

Brian C. O'Connor, Major Professor
Cathleen A. Norris, Committee Member
Sang Suh, Committee Member
Herman L. Totten, Dean of the School of Library and Information Sciences
Sandra L. Terrell, Dean of the Robert B. Toulouse School of Graduate Studies

McWhorter, William Isaac. The effectiveness of using LEGO® Mindstorms® robotics activities to influence self-regulated learning in a university introductory computer programming course. Doctor of Philosophy (Information Science), May 2008, 144 pp., 56 tables, 15 figures, references, 54 titles.

The research described in this dissertation examines the possible link between self-regulated learning and LEGO Mindstorms robotic activities in teaching concepts in an introductory university computer programming course. The areas of student motivation, learning strategies, and mastery of course objectives are investigated. In all three cases analysis failed to reveal any statistically significant differences between the traditional control group and the experimental LEGO Mindstorms group as measured by the Motivated Strategies for Learning Questionnaire and course exams. Possible reasons for the lack of positive results include technical problems and limitations of the LEGO Mindstorms systems, limited number and availability of robots outside of class, limited amount of time during the semester for the robotic activities, and a possible difference in effectiveness based on gender. Responses to student follow-up questions, however, suggest that at least some of the students really enjoyed the LEGO activities. As with any teaching tool or activity, there are numerous ways in which LEGO Mindstorms can be incorporated into learning. This study explores whether or not LEGO Mindstorms are an effective tool for teaching introductory computer programming at the university level and how these systems can best be utilized.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank several people who have supported and encouraged me throughout my doctoral experience. First, to my committee chair, Brian O'Connor, whose discussions and interest in LEGO Mindstorms systems led me to this dissertation topic, thank you for your guidance, perspective, and coffee. Thank you also to my committee members, Cathleen Norris and Sang Suh, for your knowledge and enthusiasm. Thank you to my department head at Texas A&M University – Commerce, Sam Saffer, for allowing the Computer Science Department to purchase the LEGO Mindstorms kits used in this study. To my students who participated in this study, thank you for taking the time and effort to do something that was a little different but hopefully fun and beneficial. Finally, to my parents, Bill and Ulna McWhorter, whose value of higher education was instilled in me, thank you for all your support and for being such good role models in my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The last thirty years have brought increasingly rapid changes in the world of computers. Technology keeps getting smaller and faster. Great advances have been made in both hardware and software; however, many issues of use in the 1970s remain unresolved today. One such challenge concerns teaching students to program computers. Some learners seem to have a natural ability while others struggle.

Many studies have attempted to identify factors that influence success in an introductory computer programming course. Wilson and Shrock (2001) examined several such factors including math background, attribution for success/failure, domain specific self-efficacy, encouragement, course comfort level, work style preference, previous programming experience, and gender. The results of their study suggested that course comfort level had the strongest influence on success, followed by math background. Previous training in computer programming also helped. According to Pea and Sheingold (1987), at least six factors are frequently mentioned as being cognitive prerequisites to learning programming. They are mathematical ability, memory capacity, analogical reasoning, conditional reasoning, procedural thinking, and temporal reasoning. Wilson and Shrock stress the importance of providing computer programming students with an environment that is comfortable and not intimidating so that they can ask questions and improve their programming abilities.

One recent trend is the use of programmable LEGO® Mindstorms® robots (LEGO Group, http://mindstorms.lego.com) in introductory programming courses. The technology used may be relatively new but the ideas that have influenced such activities are not. LEGO Mindstorms systems, which started as a joint venture between the LEGO® Corporation (LEGO

Group, http://www.lego.com) and the Massachusetts Institute of Technology, owe its name to Seymour Papert. Papert's book entitled *Mindstorms: Children, Computers, and Powerful Ideas* (1980) has influenced generations of teachers and students. Papert challenges readers not just to think, but also to "think about thinking" (p. 27).

Another important topic in education and psychology is self-regulated learning (SRL). Bergin, Reilly, and Traynor (2005) define self-regulated learning as "the degree to which learners are metacognitively, motivationally and behaviorally active participants in their own academic learning" (p. 81). Components of self-regulated learning can be broken down into two main areas: motivation and learning strategies (sometimes referred to as the will and skill of a student). According to Pintrinch and DeGroot (1990), the skill and will components encompassing self-regulated learning have been found to have a strong influence on the success of students in college.

The end result of any teaching technique is to improve student learning, which is traditionally measured using some kind of quantitative instrument. Most introductory computer programming courses share a number of common concepts to be learned. These include objectives such as the use of selection structures, repetition structures, functions, and arrays.

Although there have been a handful of studies that explore the effectiveness of teaching with LEGO Mindstorms robots, there exists a lack of research that attempts to examine the relationship between this teaching strategy, self-regulated learning, and the study of introductory computer programming.

Purpose Statement

The purpose of this study is to investigate the effectiveness of LEGO Mindstorms robotics activities in influencing students' motivation, learning strategies, and mastery of course objectives in an introductory university computer programming course.

Research Questions

The study centered around the following three main research questions:

- Research Question 1: How will using LEGO Mindstorms programming activities affect the motivation of students in a university introductory computer programming course?
- Research Question 2: How will using LEGO Mindstorms programming activities affect the learning strategies of students in a university introductory computer programming course?
- Research Question 3: How will using LEGO Mindstorms programming activities affect the mastery of course objectives in a university introductory computer programming course?

In order to answer research questions one and two, the Motivated Strategies for Learning Questionnaire (MSLQ) was used. This instrument, developed by Paul Pintrich, David Smith, Teresa Garcia, and Wilbert McKeachie (1991), measures various components relating to self-regulated learning. The 81 question survey was administered twice to both a control group taking part in a traditional introductory computer programming course and a group incorporating LEGO Mindstorms robotic activities. The motivational portion of the MSLQ is broken down into several sub-scales. These sub-scales consist of intrinsic goal orientation, extrinsic goal orientation, task value, control of learning beliefs, self-efficacy, and test anxiety. The learning strategies portion of the MSLQ is broken down into rehearsal, elaboration, critical thinking, metacognition, time and study environment, effort regulation, peer learning, and help seeking.

Measurement of course learning objectives was accomplished through the use of course exams. The questions on these exams came from the textbook test bank designed by D.S. Malik (2007). In order to answer research question three, percentage of answers correct on exam two (which served as a pretest) was compared to percentage of answers correct on the final exam. The same exams were used in both the control group and experimental LEGO group. The specific comparisons made were overall exam score, selection structures, repetition structures, functions, and one-dimensional arrays. Because the topic of arrays had not been covered at the time of exam two, only the results of the final exam were used to examine mastery of one-dimensional array concepts.

Each of the three major research questions was broken down into a series of hypotheses relating to each scale to be measured. Each of these hypotheses is formally introduced in chapter three of this document. An analysis of variance was performed for each hypothesis and is included in this report along with related descriptive statistics.

Definition of Terms

The study described in this document examines the interaction between LEGO Mindstorms robotic activities and self-regulated learning in an introductory university computer programming course. The definitions of terms contained in Table 1 apply to this study.

Table 1

Definition of Selected Terms

Term	Definition
Self-regulated Learning	The degree to which learners are metacognitively, motivationally, and behaviorally active participants in their own academic learning.
LEGO Mindstorms Invention System 2.0	A product from the LEGO Corporation that allows users to construct programmable robots.
Intrinsic Goal Orientation	Motivation related to the task itself. Examples include challenge, curiosity, and desire to master the material.
Extrinsic Goal Orientation	Motivation that comes from things other than the actual task. Examples include grades, rewards, and competition.
Task Value	Refers to students' perception of how important, interesting, or useful the actual task is.
Control of Learning	Refers to a student's belief that learning will lead to positive outcomes.
Self-Efficacy	Refers to students' view of their ability to learn new material and to accomplish tasks successfully.
Test Anxiety	Incorporates aspects of worry and emotion that can have a negative effect on performance.

Rehearsal Strategies	Reciting information from working memory.
Elaboration Strategies	Activities such as paraphrasing and creating analogies used to transfer information into long-term memory.
Organizational Strategies	Strategies such as outlining, clustering, and identifying main ideas.
Critical Thinking	Applying previously learned knowledge to new situations in order to solve problems.
Metacognition	Ability to set goals, maintain attention, and assimilate new ideas into things previously learned.
Time and Study Environment	Refers to the learner's ability to schedule and maintain appropriate amounts of study time in an environment that is quiet and relatively free from distractions.
Effort Regulation	The ability to maintain attention even in the presence of distractions.
Peer Learning	Activities that allow students to work together.
Help Seeking	Getting help from the instructor or a fellow student when a concept is unclear.
Selection Structure	Computer programming control structure that tests a true / false condition and takes one of two courses of action based on the result.
Repetition Structure	Computer programming control structure that tests a true / false

	condition and repeats instructions until the condition is false.
Function	A module of code that accomplishes a task. Functions are typically part of a larger program.
One-dimensional Array	A list of related data items that is given one name. In most programming languages the array items must be of the same data type.

Limitations and Delimitations

The following limitations and delimitations apply to this study.

1. The study was restricted to students taking CSCI 151: Introduction to C++ Programming during the Fall 2005 and Spring 2006 semesters at Texas A&M University - Commerce. Generalizations to other populations may not be appropriate.
2. The study was restricted to students who completed all three course exams and both pretest and posttest MSLQ. Students who dropped the course or simply stopped attending class were not included in the study.
3. The study was restricted to only those students in the experimental group participating in the LEGO Mindstorms activities.
4. All course sections included in this study were taught by William McWhorter.
5. Although the author of the textbook was contacted and stated that he believed the test bank questions to be good and reliable, specific validity and reliability statistics were not available. As a result, the power of statistical analysis pertaining to mastery of course objectives may be reduced due to a lack of a validated measuring instrument.

Organization of Remaining Chapters

Chapter two provides an overview of the theoretical background and a review of the literature that serves as the basis for the study. Chapter three describes the procedures used to conduct the experiment, analyze data, and control threats to validity. Chapter four presents the statistical results pertaining to each research question and hypothesis. Finally, chapter five provides a discussion of possible reasons for the results of the study, as well as exploring a series of follow-up interviews conducted to further examine certain questions. Conclusions of the study and possibilities for future research are also presented in chapter five.

CHAPTER 2

REVIEW OF LITERATURE

Seymour Papert, Constructionism, and Logo

Seymour Papert is known as the founding father of the Logo programming environment and the philosophy of learning which it promotes. He argued that Logo is more than just a programming language and learning environment for children. The ideals that it incorporates can be traced back to the constructivist education ideas of Jean Piaget, whom Papert worked with during the late 1950s and early 1960s. Papert stated, “The principle behind constructivism is that we learn better by doing and we learn even more if we combine our doing with talking and thinking about what we have done” (1999, p. VI). Papert also explained that the philosophy of learning at the heart of Logo is that learning is about more than simply getting right or wrong answers; learning is about life and getting things to work. This doesn’t necessarily mean that any method of implementation is fine. Often, there are many different ways of solving a problem, and some solutions are more efficient than others. The discussion of advantages and disadvantages of different solutions is a valuable part of learning problem solving techniques.

Papert (1999) also argued that there is a difference between constructivism and what he referred to as constructionism. He stated that his word, constructionism, goes beyond the “learning by doing” philosophy of constructivism and should be thought of as “learning by making.” The difference is subtle, but constructionism encompasses not only teaching of concepts but also deciding what citizens of the future need to know in order to continue constructing new ideas and technology. Papert stated:

> We learn best of all by the special kind of doing that consists of constructing something outside of ourselves: a child building a tower, writing a story,

> constructing a working robotic device or making a video game are all examples of constructing. All these activities have several features in common. They are subject to the test of reality; if they don't work, they are a challenge to understand why and to overcome the obstacles (p. XIII).

In his book *The Children's Machine* (1993), Papert referred to the African proverb that "if a man is hungry you can give him a fish, but it is better to give him a line and teach him to catch fish himself" (p. 139). He stated that constructionism works in a similar way. Children learn best by seeking out the specific knowledge they need to accomplish their goal. Teachers should be there to facilitate, guide, and provide the tools they require.

Papert (1993) illustrated his ideas with stories of children who have benefited from constructionist teaching. For example, he mentioned a girl named Maria who learned about elements of computer programming and mathematics by building a house out of LEGO® bricks (LEGO Group, http://www.lego.com). Her class was participating in learning activities using LEGO-Logo systems, which were a precursor to LEGO® Mindstorms® systems (LEGO Group, http://mindstorms.lego.com). Maria's teacher initially explained the activity by showing an example of a LEGO truck. Initially, Maria was afraid the activity was going to be more for boys, but she was later relieved to find out that she could build a house instead. Papert cited this as an example of how constructionism can be used to appeal to differences in gender. He claimed that the concept can be expanded to bridge social, ethnic, and learning style gaps. Maria's main goal initially was to make the house as beautiful as possible. Later, as she noticed other groups incorporating technology such as gears and sensors, she decided that her house should have a flashing light. In order to make the LEGO light flash the way she wanted, she needed to program it using a computer. Through trial and error and several attempts she finally got the light to flash on and off. Without realizing it Maria had learned about loops, one of the main control structures of computer programming.

Motivation and analogical reasoning are factors that can have a strong influence in determining student success in a computer programming course. Papert (1980) argued for the use of what he called syntonic learning in dealing with such factors. Syntonic learning attempts to link mental and physical components of learning so that new knowledge can be assimilated. Syntonic learning can be accomplished by designing activities that allow the student to relate and use previously learned knowledge to solve problems. Papert stated that there are different types of syntonicity that can be incorporated. He explained that Logo geometry activities, such as using a Logo turtle to draw a circle, is "body syntonic" because the circle is firmly related to children's sense and knowledge about their own bodies. Activities can also be "ego syntonic" in that they relate to learners' intentions, goals, desires, likes, and dislikes. Ego syntonicity can have a powerful influence on motivation if it produces pride and excitement for the student. Finally, cultural syntonicity can connect school activities to problems faced in an educational environment. For example, Papert explained that the concept of navigation, which is used daily in activities such as driving, boating, and flying, can be used to connect to geometric concepts such as angles. In his book, *Mindstorms* (1980), Papert referred to a story from his youth in which he had developed an interest in how gears worked. He was surprised to discover that many adults did not understand nor care about the concept of gears. He wondered "How could what was so simple for me be incomprehensible to other people?" Then he noticed that some of the same people who could not grasp such concepts were easily able to do tasks that he considered difficult. This eventually led him to conclude that "Anything is easy if you can assimilate it to your collection of models. If you can't, anything can be painfully difficult" (p. xix).

Papert (1999) stated that his Logo philosophy poses several challenges. For example, he discussed the notion that teachers and students should be thought of as co-learners. This concept can be difficult for many teachers because they already know the material or they are unwilling or too busy to take the time to invest in new learning. Another obstacle is the limitations of technology. Papert made the point that, throughout history, the technology of the day has placed limits on learning. He used the example of geometry and stated that the limitations of two-dimensional pencil and paper learning impeded the progress of powerful ideas that are the result of three-dimensional thinking. The Logo philosophy attempts to free itself from the constraints of technology and allow learners to think outside the box.

Self-Regulated Learning

As mentioned earlier, the term self-regulated learning refers to the level at which students are motivated to learn and the strategies they incorporate in order to meet their learning goals.

Motivation

Motivating students has likely been a challenge as long as there has been formal teaching. Teaching computer science concepts such as programming requires not only good teaching techniques but also the ability to effectively incorporate technical course material. Tharp (1987) discussed some of these challenges and provided computer science teachers with useful tips on ways to motivate their students. He stated that most of these tips are simply common things that most teachers already do but it is important to revisit them from time to time. The motivational factors he elaborated on can be categorized as those for interacting with students, those for communicating information to students, and those for providing an appropriate environment.

Strategies to remember when interacting with students include caring for and identifying with students. Open discussion is also important and can build confidence as well as stimulate students to think. Instructors should not ridicule, embarrass, or place the student under undue pressure. Instead, students should be provided with continual feedback and understand what is expected of them. Assisting students with problems is also necessary for effective interaction.

Communicating course material to students in the appropriate way is an important motivational factor for student learning. Instructors must make certain that learning objectives are clear and should explain how topics relate to each other. Staging the presentation of material in an enthusiastic way can be enjoyable for students and lead to more effective communication.

The third motivational factor that Tharp discussed is providing a proper environment for learning. He stated that instructors should always be prepared and strive to make the course fun. Learning should be an activity that is shared by both students and teachers and should be made as easy as possible. Tharp's motivational tips for effective computer science instruction are consistent with Wilson and Shrock's (2001) findings that stressed the importance of comfort level as a factor in determining student success in computer programming.

Jenkins (2001) discussed the importance of student motivation in the teaching of computer programming and focused on four types of motivation: extrinsic, intrinsic, social, and achievement. His study suggested that a large number of undergraduate students are extrinsically motivated, meaning that they are motivated to learn computer programming because they believe it will lead to rewards such as better career opportunities. An almost equal number of students seemed to be intrinsically motivated, or interested in the actual learning process. Jenkins pointed out, however, that these students appear to be more interested in learning in general rather than specific learning of computer programming. These results serve as an example of the challenges that programming instructors face in understanding the motivation of their students.

Anderson and McLoughlin (2007) mentioned the lack of patience that today's computer programming students exhibit and how impatience can lead to frustration and confusion. The lack of immediate results and easy success that often comes with learning programming can have a dramatic negative impact on student motivation. This frustration can lead to a downward spiral of falling behind and eventually ending in exam failure.

Learning Styles and Strategies

Various learning styles and the nature of the material being studied has a large influence on the learning strategies that students choose to incorporate. While discussing areas of research that have contributed to modern learning theory, Hodgins and Connor (2000) explained that one related area referred to as perceptual modality seeks to understand the way in which biological responses to environmental stimuli allow for gathering information about the world. These different types of stimuli can be placed into the categories of visual, auditory, and kinetic with learners usually preferring one over the others. Hodgins and Connor argued that teaching should ideally match the mode of presentation to meet the needs of all three categories.

Bloom's Taxonomy (Bloom, Mesia and Krathwohl, 1964) attempted to create a model that combines learning styles with personality types. This model categorized different types of learning as either cognitive (knowledge-based), affective (attitude-based), or psychomotor (skills-based). Different levels of learning can also take place in each category. For example, cognitive learning can be as simple as memorization of facts, but at a higher level those facts can be comprehended and incorporated into other knowledge.

Cognitive psychologist Howard Gardner (1983) is famous for proposing a theory that stated that human intelligence can be broken down into eight areas. These consist of linguistic, musical, logical, mathematical, spatial, bodily-kinesthetic, intrapersonal, and interpersonal abilities. His “Theory of Multiple Intelligences” suggested that people are capable of intelligence in each of these categories, but usually one or two of them tend to be more dominant.

Based on these and other learning style models, the difficulty of developing activities and curriculum that will appeal to all students becomes obvious. An ideal methodology used to teach computer programming to all students should appeal to each of these various styles of learning. The learning and study strategies students choose to incorporate is influenced by their preferred learning style as well as the type of course material at hand. If these various strategies could be identified and measured, the effectiveness of such methods and activities could be evaluated.

Paul Pintrich and the Motivated Strategies for Learning Questionnaire

In order to examine the concept of self-regulated learning, a model is needed that incorporates both elements of a student’s skill and will. One such model has been developed by the team of Paul Pintrich, David Smith, Teresa Garcia, and Wilbert McKeachie (1991). The skill component of Pintrich’s model includes cognitive, metacognitive, and resource management strategies. The will component measures different areas related to motivation, such as value, expectancy, and test anxiety. Levels of degree can be obtained in each of these areas through the use of the Motivated Strategies for Learning Questionnaire (MSLQ), which was developed by Pintrich’s team of researchers. The MSLQ measures responses on a seven point scale ranging from 1 (not at all true of me) to 7 (very true of me). Participants are asked to evaluate themselves by answering questions that cover five scales relating to motivation and nine scales relating to various learning strategies.

The value component of motivation is divided into intrinsic goal orientation, extrinsic goal orientation, and task value. Intrinsic goal orientation refers to students' perception of the reasons why they are engaging in a learning task. Such reasons could include challenge, curiosity, or mastery of material. Extrinsic goal orientation is similar to intrinsic goal orientation but focuses on reasons other than the purpose of the task itself, such as grades, rewards, and competition with fellow students. Task value refers to students' perception of how important, interesting, or useful the actual task is.

The expectancy component of motivation consists of two factors. Control of learning refers to the students' belief that their learning will lead to positive outcomes. Self-efficacy refers to the students' view of their ability to learn new material and to accomplish tasks successfully.

The final area of the MSLQ relating to motivation is test anxiety. Test anxiety incorporates aspects of both worry and emotion, which can have a negative effect on performance.

The learning strategies section of the MSLQ focuses on the cognitive, metacognitive, and resource management strategies of students. Examples of cognitive strategies are rehearsal, elaboration, and organization. Rehearsal is used to recite information, usually from working memory. Elaboration involves activities such as paraphrasing and creating analogies in order to transfer information into long-term memory. Organization strategies include outlining, clustering, and identifying main ideas. These cognitive strategies can be useful for students, but they lack the deeper thinking involved with metacognitive strategies such as critical thinking, planning, and monitoring. Critical thinking involves applying previously learned knowledge to new situations in order to solve problems. Planning involves goal setting and the ability to call upon previously learned knowledge in order to understand new ideas. Monitoring includes maintaining attention while reading and questioning concepts so that new knowledge can be assimilated into things previously learned. Metacognition is at the heart of Seymour Papert's arguments that students should learn to think and to understand how things work. In his book, *Mindstorms* (1980), while discussing the benefits of his Logo environment he stated that, "One does not expect anything to work at the first try. One does not judge by standards like right – you get a good grade and wrong – you get a bad grade. Rather one asks the question: How can I fix it? And to fix it one has first to understand what happened in its own terms" (p. 101).

In addition to cognition and metacognition, the MSLQ also examines resource management strategies such as time and study environment, effort regulation, peer learning, and help seeking. Time and study environment simply refers to a learner's ability to schedule and maintain appropriate amounts of study time in an environment that is quiet and relatively free from distractions. Effort regulation is concerned with a learner's ability to maintain attention even in the presence of distractions. Activities that allow students to work together involve peer learning and have been shown to have positive effects on achievement. Help seeking simply refers to getting help from an instructor or a fellow student when a concept is unclear (Pintrich et al., 1991).

Pintrich's research (Pintrich & DeGroot, 1990; Pintrich, 1999, 2000, 2002) has shown evidence of a general correlation between factors of self-regulated learning and success in the classroom. This research, however, is focused on showing that the MSLQ is useful for general use in college classes. There is no discussion of how applicable the MSLQ is for use in an introductory computer programming course. A study conducted by Susan Bergin, Ronan Reilly, and Desmond Traynor (2005) at the National University of Ireland Maynooth examines the relationship between introductory computer programming and self-regulated learning. The instrument used in their study was the same Motivated Strategies for Learning Questionnaire designed by Pintrich's team. The overall results of the study found that students who perform well in programming use more metacognitive and resource management strategies than lower performing students. Also, it was determined that students who have high levels of intrinsic motivation and task value perform better in programming and use more metacognitive and resource management strategies than students with low levels of intrinsic motivation and task value. This suggests that if teachers can find a way to motivate students to be interested in learning programming, they will be more likely to put more effort into deeper thinking and understanding of what they are doing. No evidence was found to suggest a significant correlation between extrinsic motivation and performance, which suggests that grades are probably not the most effective way of motivating students to learn programming. Also, the study found that the use of cognitive learning strategies such as rehearsal, elaboration, and organization are not significant factors in determining success in a computer programming course. Metacognitive and resource management strategies, however, are important. This is consistent with the Pintrich et al. (1991) claim that cognitive strategies, such as rehearsal, are useful in activation of information in short term memory but lack the ability to construct internal connections that are used to build upon prior knowledge and to transfer new knowledge into long-term memory.

The results of the Bergin et al. (2005) study are important because they show that Pintrich's model of self-regulated learning and the Motivated Strategies for Learning Questionnaire are applicable to students in an introductory programming course. The study also supports the importance of factors such as intrinsic motivation, task value, metacognitive strategies, and resource management strategies in such a course. Bergin stated that, “It would appear that specifically designed tools that help students to self-regulate their learning and to encourage students to develop an intrinsic goal orientation and higher task values might enable them to achieve higher results and to promote their SRL development” (p. 85). The following section presents a survey of programming tools and environments that have been developed in an attempt to aid novices in the learning of computer programming.

Tools and Environments for Teaching Introductory Computer Programming

The search for a method of teaching computer programming in a way that is interesting and motivating to students is nothing new. Computer science education literature is full of articles describing the use of various tools, environments, and programming languages that hope to prove effective in college courses. Most fall into the categories of narrative tools, graphical and coding practice tools, approaches that use alternative languages, or assignments that attempt to gear themselves more towards the interests of students.

One of the earliest narrative tools was developed by Richard Pattis (1995). His landmark book titled *Karel the Robot: A Gentle Introduction to the Art of Programming* introduced learners to a robot named Karel. By issuing instructions, students could help Karel navigate and interact with objects in his world. By incorporating a software simulator, the effect of Karel’s movements could be observed and instructions could be modified if needed. The basic idea of Karel the Robot can still be seen in recently developed tools. For example, Anderson and McLoughlin (2007) describe their tool named C-Sheep as a re-imagination of Pattis’ original Karel. C-Sheep uses a game engine and a 3D graphics engine to allow learners to manipulate the environment of a three dimensional virtual world. This approach is also utilized in a tool called

MUPPETS (Phelps, Bierre, & Parks, 2003), or Multi-User Programming Pedagogy for Enhancing Traditional Study, as well as ALICE (Cooper, Dann, & Pausch, 2000), which has become a popular alternative to traditional programming instruction. All of these narrative tools present the learner with what Papert refers to as Microworlds. Papert (1980) defines Microworlds as computer generated environments that promote student problem solving skills by providing the tools, structures, and activities that represent a domain of mathematics and science.

Some programming tools have been developed to help free novices from the rigidity and frustration of learning program syntax. Some of these environments allow students to practice their coding skills with smaller segments of code, while others provide a more visual approach. Code Lab, developed by Turing's Craft (n.d.), has been adopted by numerous colleges and universities as a way of offering Web based coding practice exercises to computer science students and instructors. BlueJ (n.d.) is a Java Integrated Development Environment that has become a popular tool for teaching an objects-first paradigm to introductory students. It incorporates both traditional text instructions and graphics to provide learners with a better understanding of their code. Another tool called Jeliot (n.d.) goes one step further by offering visual animations that show how individual instructions affect users' programs. One older and proven approach to teaching programming concepts and algorithm design is through the use of flowcharts. A tool called RAPTOR (Carlisle, Wilson, Humphries, & Moore, n.d.) has become a popular development environment that allows students to drag and drop different flow chart symbols to construct a program algorithm.

Some computer science instructors have focused on the choice of specific programming language and assignments in an attempt to motivate students. Scheme, a more modern dialect of the LISP language, has been adopted by some universities because it utilizes simple and flexible syntax. This approach can offer a less intimidating environment, especially for non computer science students and at liberal arts colleges (Konstam & Howland, 1994). Some instructors have attempted to reach students by appealing to perceived interests of young people. Introductory programming courses that incorporate the development of games have become more and more common (Leutenegger & Edgington, 2007). The choice of JavaScript as a programming language has also been documented as a way of combining introductory programming concepts with the Web development interests of students (Mahmoud, Dobosiewicz, & Swayne, 2004).

All of the previously described tools and environments share one thing in common; they have been developed and tried as possible ways to provide introductory computer programming students with a more friendly and effective approach to learning to program. However, they are all limited in the same way. They confine learners to the constraints of a computer screen. There is currently only one widely accepted, documented, and affordable approach that teaches computer programming concepts using physical real world systems; that is LEGO Mindstorms.

LEGO Mindstorms

The LEGO Mindstorms Invention System 2.0, which gets its name from Papert's (1980) groundbreaking book, is a product from the LEGO Corporation that allows users to construct programmable robots. The robots are built of conventional LEGO parts attached to a programmable LEGO brick called the RCX. The RCX contains three input ports and three output ports attached to a Hitachi H8/3292 micro controller. Included in the set are two touch sensors, one light sensor, and two motors. Additional sensors, including rotation and temperature sensors, as well as additional motors and gears, are also available for purchase.

When building a LEGO Mindstorms robot, users go through a series of four main steps. First, they construct the physical robot. Next, using a computer, they write program instructions for the robot to execute. These programs can be written in one of the various programming interfaces available to work with the RCX brick. After the program has been written, it is transferred to the RCX brick using an infrared transmitter attached to the computer. Finally, the RCX brick, which controls the actions of the robot, executes the program. It is quite common for the last three steps to be repeated after necessary modifications are done in order to get the robot to successfully accomplish all the actions desired (McNeill & Binkerd, 2001).

Several programming interfaces are available for the LEGO Mindstorms Invention System 2.0. The kit comes with a graphical programming interface usually referred to as RCX Code. There are, however, several limitations to the native RCX code. For example, the possibility of using variables is practically non-existent with the native LEGO programming interface. In addition, subroutines cannot call other subroutines. Support for these concepts is needed if LEGO robots are to be used in an introductory computer programming course. One alternative to RCX code is the Not Quite C (NQC) programming language. It was developed by Dave Baum (2000) to provide a more powerful and traditional text-based programming environment. NQC is designed to resemble the popular C computer programming language. Other programming interfaces are also available for the LEGO Mindstorms Invention System, including the Java based leJOS, ADA, Programmable Brick FORTH, and Bot-Kit, which incorporates the Smalltalk programming language. One additional interface of interest comes bundled with the LEGO Robolab System, a special version of LEGO Mindstorms developed for educators. The Robolab software, powered by LABVIEW, provides a more powerful graphical programming environment than LEGO's RCX code (McNeill & Binkerd, 2001).

Despite being developed with teenagers in mind, the use of LEGO Mindstorms in the college computer science curriculum has attracted attention. Klassner and Anderson (2003) presented an argument for the use of LEGO Mindstorms activities to augment course content throughout the entire core of an undergraduate computer science program. They chose the LEGO Mindstorms platform over other similar robotics systems for four main reasons: cost, flexibility, student interest, and professional curiosity. They provided a brief description of how LEGO Mindstorms could be used in different areas of typical computer science core classes, including programming fundamentals, algorithms and complexity, programming languages, computer architecture, operating systems, artificial intelligence, and net-centric computing. Klassner and Anderson pointed out some of the limitations of the LEGO Mindstorms kits but discussed how these limitations can be addressed through the use of various programming languages and environments, including the "More Than Mindstorms," or MTM Tool Set, which they created.

Cliburn (2006) described how LEGO Mindstorms were used in an introductory computer science course for non majors at Hanover College. Hanover College is a liberal arts college and the course discussed provides a breadth-first overview of the field of computer science. LEGO Mindstorms robotic challenges were used to introduce students to abstraction, algorithms, and problem solving. Because the course was not intended to be a rigorous programming course, Cliburn chose to utilize the simpler and more visual programming interface included with the LEGO Mindstorms software. This allowed students to focus on the problem solving side of things rather than having to learn the syntax of a higher level language. He mentioned that most of the students seemed to really enjoy the LEGO projects and recommended their use as a tool to teach algorithms and foster student creativity.

Garcia and McNeill (2002) utilized LEGO Mindstorms activities at Texas A&M University – Corpus Christi to teach Systems Analysis and Design. Groups of students were put in charge of planning, designing, and implementing LEGO robots, which were used in a competition at the end of the semester. Models and diagrams such as GANT graphs, PERT charts, data flow diagrams, use cases, entity relationship diagrams, and state transition diagrams were included in the project requirements. Garcia and McNeill argued that the motivational factors associated with LEGO Mindstorms and the project competition made learning more fun and allowed students to control and manipulate computers in the real world.

Although journal articles have shown that LEGO Mindstorms activities can be utilized throughout the entire computer science curriculum, their most popular use in the college environment has been to help teach introductory computer programming concepts. A group of professors led by Lawhead, Duncan, Bland, Goldweber, Schep, and Barnes (2002) argued for the use of LEGO robots to teach introductory computer programming and provided tips and examples on how to use them. They explained that the use of robots exposes students to a rich and real world environment that encourages experimentation and can appeal to minorities and women. The group of researchers introduced the idea of presenting the robot as a real physical object in order to teach concepts of object-oriented programming. Unlike in a traditional programming environment, the operations or methods of a LEGO robot can be physically observed. This can make the normally boring and frustrating task of debugging more fun for students. One member of the group, David Barnes (2002), provided an overview of the pros and cons of using LEGO Mindstorms to teach introductory Java. He reported that issues such as motor voltage inconsistencies, possible confusing use of loop structures, and the need to explain the more advanced concept of event driven programming makes the use of LEGO Mindstorms impractical for teaching an entire introductory programming course. Instead he argued that the use of these physical models should be used to enhance and support concepts in a traditional course setting.

Some authors have argued that perceived problems with LEGO Mindstorms can be used in a positive way. Wolz (2001) utilized a LEGO Mindstorms project to teach students the importance of software planning and project management at the College of New Jersey. Her three week activity called “Robot Planning and Design” placed students in a highly restrictive environment that forced them to thoroughly think about a software solution without having the ability to do a lot of hands-on debugging. Students working in groups had to design and write a program to navigate a LEGO robot through a maze. The maze was not made available until the day of demonstration and the students had limited access to the actual robot. The intent of the activity was to focus on the importance of good problem solving and planning. Although none of the groups was completely successful in their attempts, the robot demonstrations served as a fun learning experience and reemphasized the need for proper planning and thinking involved in problem solving.

Despite the increased use of LEGO Mindstorms in introductory computer programming classes, very few studies have been published that examine the effectiveness of their use. One very limited study was conducted by Ka-Wing Wong (2001) at Eastern Kentucky University. Wong believed that the use of LEGO Mindstorms robot activities could provide an environment that was more effective and motivational than the traditional Integrated Development Environments used in most computer programming courses. He incorporated three week LEGO Mindstorms activities into introductory, intermediate, and advanced computer science courses. Wong claimed that the students seemed to retain learned knowledge better in the LEGO sections than non LEGO sections. Students in the LEGO introductory programming course scored 2% higher on a course test given after the activities when compared to a normal section. The results were even better at the intermediate data structures course level where students reported a 4% higher average. Students also reported on a survey that they preferred learning programming through the LEGO robotics activities. Although the findings of Wong's study were positive, it should be noted that there were only 17 students in the introductory programming course and 11 in intermediate data structures.

The largest related study attempting to measure the effects of LEGO Mindstorms activities in an introductory computer programming course was performed by Fagin and Merkle (2002) at the United States Air Force Academy during the fall 2000/spring 2001 school year. The focus of the study was to see if using LEGO Mindstorms improved student performance on identical exams. The study also looked at whether students in the LEGO group were more likely to choose computer science as a major. The results of the study were negative. Test scores in the robotics sections were lower than the non-robotics sections. Also, the use of robots did not show any measurable effects on choice of major. Because introductory computer programming is required by all students at the Air Force Academy, the study was able to incorporate a large sample. The course was taught to 938 students in 48 sections of 15-20 students each. LEGO robotics activities were performed in nine of the sections. The programming language used by the students was ADA, which was translated into Not Quite C using software developed by Fagin's team. The effects of GPA were removed from students' raw exam scores using linear regression, and the Kruskal-Wallis H test was used to examine differences on the exams between groups. The results showed that there was a statistically significant difference between groups based on their performance on all three exams during the academic year and that the LEGO groups performed worse than the groups that went through regular lab activities. The *p* value from the Kruskal-Wallis test after removal of GPA was less than .005 on exams one and two, and was .01 on the final exam.

The Fisher-Irwin test was used to examine the relationship between groups in relation to selection of majors. Students at the Air Force Academy are required to choose a major no later than the second semester of their sophomore year. The researchers wanted to know if students in the robotics sections were more likely to choose either computer science or computer engineering as a major. No statistically significant results were found between groups. "Over the course of the complete academic year, the cadets in the robotics sections were slightly less likely to eventually declare the computer science major" (p. 8).

While the Fagin study did not look closely at the issue of motivation, there were a few related questions on the end of semester course evaluations. A scale of 1 through 5 was used to record student responses. Students in the non-robotics sections rated the course slightly higher than the robotics sections in relation to the course as a whole, relevance and usefulness, amount learned, and instructor effectiveness.

Given the results of this study, it becomes questionable whether LEGO Mindstorms robotics activities are appropriate in an introductory computer programming class; yet, their use has become more and more common among colleges and universities. A major reason for choosing to incorporate LEGO Mindstorms is the belief that it helps motivate students, yet there is no major study that attempts to quantify such effects.

CHAPTER 3

METHOD OF PROCEDURE

The following section describes the design, research questions and hypotheses, participants, measurement instruments, experimental procedures, and data analysis procedures of this research study. Possible threats to validity are also discussed.

Design of Study

This research study utilized data taken from students enrolled in CSCI 151: Introduction to C++ Programming during the 2005-2006 academic school year at Texas A&M University - Commerce. The research design for the study was a quasi-experimental pretest/posttest design for the dependent variables, using an experimental group of introductory programming students taking part in LEGO® Mindstorms® systems (LEGO Group, http://mindstorms.lego.com) class activities and a group of students taking part in a traditional introductory programming course. The control group was comprised of two sections of CSCI 151 during the Fall 2005 semester. The experimental LEGO Mindstorms group was comprised of two sections of CSCI 151 during the Spring 2006 semester. The two groups were divided by semester, both for convenience and to attempt to control for possible feelings of jealousy that the traditional group might have towards the LEGO group. All students were included in the study with the exception of students who dropped the course and students who stopped coming to class and did not take the final exam.

The study looked for statistical differences between groups in each scale of Pintrich's self-regulated learning model as well as performance on major course objectives. A summary of each item measured is shown in Table 2.

Table 2

Summary of Measured Items

Item to Be Measured	Definition
Intrinsic Goal Orientation	Motivation related to the task itself. Examples include challenge, curiosity, and desire to master the material.
Extrinsic Goal Orientation	Motivation that comes from things other than the actual task. Examples include grades, rewards, and competition.
Task Value	Refers to the student's perception of how important, interesting, or useful the actual task is.
Control of Learning	Refers to a student's belief that learning will lead to positive outcomes.
Self-efficacy	Refers to a student's view of his ability to learn new material and to accomplish tasks successfully.
Test Anxiety	Incorporates aspects of worry and emotion that can have a negative effect on performance.
Rehearsal Strategies	Reciting information from working memory.
Elaboration Strategies	Activities such as paraphrasing and creating analogies used to transfer information into long-term memory.
Organizational Strategies	Strategies such as outlining, clustering, and identifying main ideas.
Critical Thinking Strategies	Applying previously learned knowledge to new situations in

	order to solve problems.
Metacognitive Strategies	Ability to set goals, maintain attention, and assimilate new ideas into things previously learned.
Time and Study Environment Strategies	Refers to the learner's ability to schedule and maintain appropriate amounts of study time in an environment that is quiet and relatively free from distractions.
Effort Regulation Strategies	The ability to maintain attention even in the presence of distractions.
Peer Learning Strategies	Used during activities that allow students to work together.
Help Seeking Strategies	Getting help from the instructor or a fellow student when a concept is unclear.
Overall Final Exam Score	The final exam contains a combination of multiple choice and true/false questions.
Understanding of Selection Structures	Computer programming control structure that tests a true / false condition and takes one of two courses of action based on the result.
Understanding of Repetition Structures	Computer programming control structure that tests a true / false condition and repeats instructions until the condition is false.
Understanding of Functions	A module of code that accomplishes a task. Functions are typically part of a larger program.
Understanding of One Dimensional Arrays	A list of related data items that is given one name. In most programming languages, including C++, the array items must be of the same data type.

Research Questions and Hypotheses

The research study focused on three major research questions with each question having a number of hypotheses that were tested. They are as follows:

- Research Question 1: How will using LEGO Mindstorms programming activities affect the motivation of students in a university introductory computer programming course?
    - Hypothesis 1: Students taking part in LEGO Mindstorms programming activities will show higher levels of intrinsic motivation in relation to their programming course than students taking part in a traditional computer programming course.
    - Hypothesis 2: Students taking part in LEGO Mindstorms programming activities will show higher levels of extrinsic motivation in relation to their programming course than students taking part in a traditional computer programming course.
    - Hypothesis 3: Students taking part in LEGO Mindstorms programming activities will show higher levels of task value in relation to their programming course than students taking part in a traditional computer programming course.
    - Hypothesis 4: Students taking part in LEGO Mindstorms programming activities will show higher levels of control of learning belief in relation to their programming course than students taking part in a traditional computer programming course.
    - Hypothesis 5: Students taking part in LEGO Mindstorms programming activities will show higher levels of self-efficacy in relation to their programming course than students taking part in a traditional computer programming course.

    - Hypothesis 6: Students taking part in LEGO Mindstorms programming activities will show lower levels of test anxiety in relation to their programming course than students taking part in a traditional computer programming course.

- Research Question 2: How will using LEGO Mindstorms programming activities affect the learning strategies of students in a university introductory computer programming course?
    - Hypothesis 7: Students taking part in LEGO Mindstorms programming activities will use more rehearsal strategies than students taking a traditional programming course.
    - Hypothesis 8: Students taking part in LEGO Mindstorms programming activities will use more elaboration strategies than students taking a traditional programming course.
    - Hypothesis 9: Students taking part in LEGO Mindstorms programming activities will use more organizational strategies than students taking a traditional programming course.
    - Hypothesis 10: Students taking part in LEGO Mindstorms programming activities will show higher levels of critical thinking strategies than students taking a traditional programming course.
    - Hypothesis 11: Students taking part in LEGO Mindstorms programming activities will show higher levels of metacognitive self-regulation strategies than students taking a traditional programming course.

    - Hypothesis 12: Students taking part in LEGO Mindstorms programming activities will show better time and study environment strategies than students taking a traditional programming course.
    - Hypothesis 13: Students taking part in LEGO Mindstorms programming activities will show better effort regulation strategies than students taking a traditional programming course.
    - Hypothesis 14: Students taking part in LEGO Mindstorms programming activities will show higher levels of peer learning strategies than students taking a traditional programming course.
    - Hypothesis 15: Students taking part in LEGO Mindstorms programming activities will show higher levels of help seeking strategies than students taking a traditional programming course.
- Research Question 3: How will using LEGO Mindstorms programming activities affect the mastery of course objectives in a university introductory computer programming course?
    - Hypothesis 16: Students taking part in LEGO Mindstorms programming activities will perform significantly better on the course final exam than students taking a traditional programming course.
    - Hypothesis 17: Students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of selection structures on the final course exam than students taking a traditional programming course.

- Hypothesis 18: Students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of repetition structures on the final course exam than students taking a traditional programming course.
- Hypothesis 19: Students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of functions on the final course exam than students taking a traditional programming course.
- Hypothesis 20: Students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of one-dimensional arrays on the final course exam than students taking a traditional programming course.

Description of Research Participants

The control group consisted of 40 students enrolled in two sections of CSCI 151 at Texas A&M University – Commerce during the Fall 2005 semester. The group was comprised of 31 males and 9 females. The ethnic makeup consisted of 9 African-American, 1 Asian-American, 25 Caucasian, 4 Hispanic, and 1 reporting as other. There were 11 freshmen, 9 sophomores, 18 juniors, and 3 seniors enrolled.

There were 43 participants in the experimental LEGO Mindstorms group. These students were taken from two sections of CSCI 151 at Texas A&M University – Commerce during the Spring 2006 semester. 35 were male and 8 were female. There were 7 African-American, 28 Caucasian, 6 Hispanic, and 2 reporting as other. The classifications were comprised of 8 freshmen, 16 sophomores, 14 juniors, and 5 seniors.

The most common majors for students in both groups were Computer Science, Computer Information Systems, Mathematics, and Industrial Engineering. There were also students majoring in Business Administration, Physics, Psychology, Sociology, Political Science, Radio/Television, Chemistry, Biology, Finance, and Health.

The purpose and procedures of the experiment were explained to all participants and each student signed a research consent form, which was approved by the Internal Review Board at both the University of North Texas and Texas A&M University – Commerce.

Description of Activities, Instruments and Data Collection Procedures

The Motivated Strategies for Learning Questionnaire (Pintrich et al., 1991) was used to measure motivation and the use of learning strategies. The research done by Bergin et al. (2005) showed that the MSQL is pertinent, valid, and reliable for use in an introductory computer programming course. The MSQL was given twice in both the Fall 2005 and Spring 2006 semesters. The first administered MSQL was on the day of exam two during both semesters and served as a pretest that was used to compare the initial levels of motivation and use of learning strategies between groups. The second MSQL was administered on the day of the final exam and served as the posttest. Students in all sections received seven extra credit points on exam two and the final exam for completing the MSQL survey.

All sections during the Spring 2006 semester participated in a three-week sequence of lab activities using LEGO Mindstorms robots. Construction of the robots was done outside of class time during a previous semester. The design of the LEGO Roverbot, as seen in Figure 1, was based on the construction instructions found in the LEGO Constructopedia™ (LEGO Group, 1999). The Not Quite C language was used for programming the LEGO robots, and the C++ language was used for a program assignment that analyzed data collected by the robots. Students worked through a Not Quite C tutorial that had been compiled using examples from Overmars

(1999) and Baum (2000). The software used during the LEGO lab sessions was *Brick Command Center* (2002). The lab environment consisted of approximately 20 computers on tables. Groups of 2-3 students shared each of the 10 LEGO Mindstorms robots that were utilized. After completion of the LEGO Mindstorms tutorial, students were asked to complete two programming assignments (Appendix D). All sections during the Fall 2005 semester participated in traditional course lectures and lab activities using C++. Microsoft Excel® spreadsheet software (Microsoft Corporation, http://office.microsoft.com/excel/) was used to compile MSLQ data in order to obtain scale results for each student. These results were then statistically analyzed using Statistical Package for the Social Sciences (SPSS®) software (SPSS Inc., http://www.spss.com).

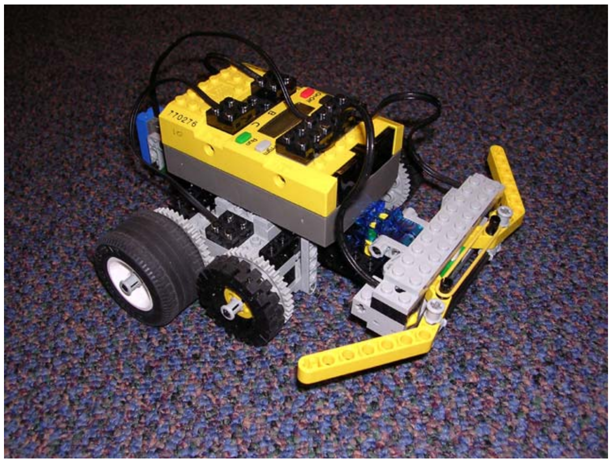


Figure 1: Photograph of a LEGO Roverbot.

In addition to the MSLQ, data taken from course exams was used to compare the effect of using LEGO Mindstorms programming activities on mastery of course objectives. Three major exams were given during the Fall 2005 and Spring 2006 semesters. Exams during the two semesters contained a combination of true/false and multiple-choice questions, which were taken from the course book test bank. Course sections of both semesters were given the same exams. After the exams were completed and graded, time was taken during class to review the questions in order to let students see what they missed. All exam materials were then recollected in order to maintain confidentiality of the exact exam contents. The LEGO Mindstorms activities were performed after exam two during the Spring 2006 semester. Sections during the Fall 2005 semester continued with traditional classroom instruction and C++ lab assignments. Exam two was used to collect pretest data while the final exam was used to collect posttest data relating to mastery of course objectives. The exams were designed to measure mastery of programming concepts such as selection structures, repetition structures, and functions. Only the final exam contained questions regarding one-dimensional arrays. A C++ program was written in order to determine each student's overall score percentage as well as percentages on each sub-scale. These results were then entered into SPSS. Because exam two and the final exam were not identical, the raw percentages were first converted into z-scores. According to Gall, Gall, and Borg (1996), a z-score is a standard score that is frequently used in educational research. It allows a person's relative standing on two or more tests to be compared because their z-scores are continuous and have equality of units. Z-scores have a mean of zero and a standard deviation of one.

The main focus of the study was the effect that LEGO Mindstorms activities have on motivation and learning strategies of students; however, the ultimate goal of any college course is to improve student learning which is traditionally measured using exams. Therefore, information concerning the mastery of course objectives was included in the study. The results obtained from evaluating exam data, however, should be viewed with caution due to the lack of a validated standardized test. The author of the textbook and test bank questions was contacted but unfortunately no reliability or validity statistics were available. Even in the absence of a validated standardized test, it was determined that the use of the textbook test bank questions was acceptable based on the cognitive authority of the author, common use in a normal course environment, and the fact that mastery of course objectives was not the primary aspect of the study.

Data Analysis Procedures

According to Gall et al. (1996), the two most common approaches for analyzing gain scores in a quasi-experimental study such as this are an analysis of covariance, or ANCOVA, and a two-way analysis of variance for repeated measures. When choosing ANCOVA, posttest levels are adjusted using pretest scores as the covariant. According to Hinkle, Wiersma, and Jurs (2003), the assumptions underlying the use of ANCOVA are a linear relationship between the dependent variable and the covariant, and homogeneity of regression between their slopes. These assumptions are in addition to the assumptions of random assignment, normal distribution, and homogeneity of variance required by ANOVA.

The other option mentioned by Gall et al. (1996) is an analysis of variance for repeated measures. Gall stated:

> This statistical technique is used to determine whether the pretest-posttest difference for the experimental group is reliably different from the pretest-posttest difference for the control group. The occasions on which the measure of the dependent variable is administered (pretest and posttest) are considered one factor, and the experimental and control treatments are the other factor. The *F* ratios for the two factors (sometimes called main effects) are not of interest in this analysis of variance. Of interest instead is the interaction between time of measurement and treatment. That is, we are interested in

> whether the difference between the pretest and posttest means of the experimental group is significantly greater or less than the difference for the control group (p. 536).

After consideration of the required assumptions of both described statistical tests, it was determined that a two-way analysis of variance for repeated measures would be the more appropriate choice for comparison of groups.

Threats to Internal Validity

Gall et al. (1996) discussed several possible threats to the internal validity of an experiment that should be dealt with in order to yield meaningful results. First, the issue of history refers to the greater chance that other events can affect results if the experimental treatment takes place over a longer period of time. In an attempt to minimize this, the LEGO Mindstorms activities in the study lasted for a three week period instead of spreading the labs throughout the entire semester. Also, there was a relatively short period of time between the pretest and posttest occasions.

In some experiments it is possible that research participants show physical or psychological maturation that can have an effect on the experiment. Physical changes are not a major concern at the college level, but it is possible that some students, such as incoming college freshmen, might become more or less mature in their study habits in college. This was minimized in the study by collecting pretest data from both the control and experimental groups and by limiting the time between the pretest and the posttest.

Another possible threat to internal validity is that participants can become test-wise from their experience with the pretest, which can have an effect on their posttest data. The MSLQ is designed to measure attitudes and opinions with many questions. It would be difficult for students to remember the exact rating they reported on each pretest question. As far as the final exam, part of the goal of a college course is that students continue to learn throughout the entire semester. One would hope that in a college course environment, students would have a better understanding of concepts on the final exam than they did on earlier exams.

Experimental mortality, or attrition, is an issue that occurs when a researcher loses research participants. In this study, data from students who dropped the course or stopped coming to class were removed due to lack of posttest results. The total number lost in the study was 23% of the control group and 24% of the experimental LEGO group.

Experimental treatment diffusion and compensatory rivalry are two related issues that can have a possible effect on internal validity. Experimental treatment diffusion refers to the possibility that participants in the control group might become aware of the experimental treatment group’s activities and seek to either learn more or become involved especially if they view the treatment as highly desirable. Compensatory rivalry refers to the possibility that the control group might feel that they are in competition with the experimental group and as a result will attempt to perform at a higher level than they would otherwise. In order to minimize these conditions, all sections constituting the control group were taken from the Fall 2005 semester and the participants were not made fully aware of the following semester’s LEGO Mindstorms activities until the end of the semester. All sections of the Spring 2006 semester underwent the LEGO experimental treatment.

CHAPTER 4

ANALYSIS AND PRESENTATION OF RESULTS

As mentioned earlier, the focus of this study was organized into three major research questions relating to the use of LEGO® Mindstorms® systems (LEGO Group, http://mindstorms.lego.com). They are as follows:

- Research Question 1: How will using LEGO Mindstorms programming activities affect the motivation of students in a university introductory computer programming course?
- Research Question 2: How will using LEGO Mindstorms programming activities affect the learning strategies of students in a university introductory computer programming course?
- Research Question 3: How will using LEGO Mindstorms programming activities affect the mastery of course objectives in a university introductory computer programming course?

Each research question was evaluated using a number of individual hypotheses. Research questions one and two used the Motivated Strategies for Learning Questionnaire described earlier. This instrument measures fifteen different scales related to motivation and learning strategies. Table 3 contains a list of these scales and the abbreviations that will be used in this section where table space is limited.

Reliability coefficients were computed from the collected student surveys using Cronbach's coefficient alpha. Gall et al. (1996) stated that, "Cronbach's alpha is a widely used method for computing test score reliability" (p. 257). The high reliability coefficients, seen in Table 4, were consistent with Pintrich's findings and indicate that the MSLQ provided reliable results on the measured scales.

Table 3

Abbreviations for MSLQ Scales

Abbreviation	Scale
I.G.O	Intrinsic Goal Orientation
E.G.O	Extrinsic Goal Orientation
T.V.	Task Value
C.L.B.	Control of Learning Beliefs
S.E.	Self-efficacy
T.A.	Test Anxiety
Reh.	Rehearsal Strategies
Elab.	Elaboration Strategies
Org.	Organizational Strategies
C.T.	Critical Thinking Strategies
M.C.	Metacognitive Strategies
T.S.E.	Time and Study Environment Strategies
E.R.	Effort Regulation Strategies
P.L.	Peer Learning Strategies
H.S.	Help Seeking Strategies

Table 4

Reliability Analysis using Cronbach Alpha Measure

	Pretest			Posttest			Pintrich
Scale	Control	LEGO	Combined	Control	LEGO	Combined	Total
	n = 38	*n* = 40	*n* = 78	*n* = 38	*n* = 40	*n* = 78	*n* = 380
1. I.G.O	.777	.804	.795	.786	.864	.833	.74
2. E.G.O	.695	.714	.707	.627	.860	.808	.62
3. T.V.	.905	.945	.933	.949	.957	.954	.90
4. C.L.B.	.818	.764	.788	.826	.723	.776	.68
5. S.E.	.891	.944	.926	.943	.937	.941	.93
6. T.A.	.818	.630	.736	.844	.829	.836	.80
7. Reh.	.750	.591	.674	.842	.716	.778	.69
8. Elab.	.767	.802	.783	.851	.851	.850	.76
9. Org.	.736	.788	.759	.860	.751	.793	.64
10. C.T.	.803	.897	.861	.869	.906	.891	.80
11. M.C.	.742	.833	.801	.861	.905	.885	.79
12. T.S.E.	.671	.847	.787	.832	.841	.830	.76
13. E.R.	.623	.786	.733	.582	.817	.740	.69
14. P.L.	.704	.700	.716	.695	.687	.694	.76
15. H.S.	.695	.805	.757	.643	.699	.671	.52

LEGO Mindstorms Effect on Student Motivation

The first research question of this study was "How will using LEGO Mindstorms programming activities affect the motivation of students in a university introductory computer programming course?" To answer this question, six hypotheses were tested using the Motivated Strategies for Learning Questionnaire. Each of these hypotheses are presented separately and accompanied by supporting statistical tables.

Hypothesis 1 states that students taking part in LEGO Mindstorms programming activities will show higher levels of intrinsic motivation in relation to their programming course than students taking part in a traditional computer programming course. Table 5 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 5

Mean Levels of Intrinsic Goal Orientation

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	4.90	1.11	38	4.92	1.25	38
LEGO	4.61	1.47	40	4.63	1.47	40
Combined	4.75	1.31	78	4.77	1.37	78

The data were analyzed using a two-way analysis of variance for repeated measures. Table 6 shows that there was no statistically significant difference between groups in relation to levels of intrinsic goal orientation (F(1,76)=.001, *p*>.05). In fact, the experimental LEGO group reported lower pretest and posttest means than the control group. These findings suggest that using LEGO

Mindstorms activities did not increase students' motivation based on interest in the material compared to a normal programming course structure. The data do not support Hypothesis 1.

Table 6

Analysis of Variance Results for Group and Intrinsic Goal Orientation

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	3.347	1	3.347	1.109	.296
Error 1	229.454	76	3.019		
			Within subjects		
IGO	.021	1	.021	.036	.850
IGO X Group	.000	1	.000	.001	.979
Error 2	43.324	76	.570		

Hypothesis 2 states that students taking part in LEGO Mindstorms programming activities will show higher levels of extrinsic motivation in relation to their programming course than students taking part in a traditional computer programming course. Table 7 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 7

Mean Levels of Extrinsic Goal Orientation

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	5.13	1.15	38	5.34	1.07	38
LEGO	5.00	1.43	40	4.59	1.73	40
Combined	5.06	1.29	78	4.95	1.49	78

Box's test of equality of covariance indicated the observed levels of the dependent variables were not equal across groups. The Greenhouse-Geisser technique was used to compensate for this. A two-way analysis of variance for repeated measures showed a statistically significant interaction effect between groups (*F*(1,76)=7.935, *p*<.05). Table 8 contains this information. The results, however, were negative. Students who participated in LEGO robotics activities exhibited a decrease in extrinsic motivation, while students who received traditional course instruction exhibited an increase. This suggests that students in the control group became more motivated because they wanted a good grade. Students in the LEGO group, however, were less motivated in the course by interests in rewards such as grades after participating in LEGO robotics activities. Even though there was a statistically significant difference in relation to extrinsic goal orientation, Hypothesis 2 states that students from the LEGO group will show higher levels on this scale. The data do not support Hypothesis 2.

Table 8

Analysis of Variance Results for Group and Extrinsic Goal Orientation

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	7.539	1	7.539	2.271	.136
Error 1	252.233	76	3.319		
Within subjects					
EGO	.424	1	.424	.908	.344
EGO X Group	3.703	1	3.703	7.935	.006
Error 2	35.463	76	.467		

Hypothesis 3 states that students taking part in LEGO Mindstorms programming activities will show higher levels of task value in relation to their programming course than students taking part in a traditional computer programming course. Table 9 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 9

Mean Levels of Task Value

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	5.32	1.25	38	5.31	1.46	38
LEGO	4.79	1.80	40	4.81	1.78	40
Combined	5.05	1.57	78	5.05	1.64	78

The data was analyzed using a two-way analysis of variance for repeated measures. Table 10 shows that there was no statistically significant difference between groups in relation to levels of task value (F(1,76)=.022, p>.05). The data do not support Hypothesis 3.

Table 10

Analysis of Variance Results for Group and Task Value

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	10.281	1	10.281	2.193	.143
Error 1	356.265	76	4.688		
Within subjects					
TV	.000	1	.000	.000	.986
TV X Group	.009	1	.009	.022	.883
Error 2	30.253	76	.398		

Hypothesis 4 states that students taking part in LEGO Mindstorms programming activities will show higher levels of control of learning belief in relation to their programming course than students taking part in a traditional computer programming course. Table 11 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 11

Mean Levels of Control of Learning Beliefs

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	5.65	1.06	38	5.74	1.14	38
LEGO	5.78	1.11	40	5.56	1.06	40
Combined	5.72	1.08	78	5.64	1.10	78

The data was analyzed using a two-way analysis of variance for repeated measures. Table 12 shows that there was no statistically significant difference between groups in relation to students' perceived control of learning beliefs ($F(1,76)=1.907$, $p>.05$). The data do not support Hypothesis 4.

Table 12

Analysis of Variance Results for Group and Control of Learning Beliefs

Source	SS	df	MS	*F*	*p*
		Between subjects			
Group	.025	1	.025	.013	.909
Error 1	144.203	76	1.897		
		Within subjects			
CLB	.190	1	.190	.385	.537
CLB X Group	.940	1	.940	1.907	.171
Error 2	37.442	76	.493		

Hypothesis 5 states that students taking part in LEGO Mindstorms programming activities will show higher levels of self-efficacy in relation to their programming course than students taking part in a traditional computer programming course. Table 13 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 13

Mean Levels of Self-Efficacy

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	5.28	1.05	38	5.46	1.26	38
LEGO	5.04	1.48	40	4.89	1.54	40
Combined	5.15	1.29	78	5.17	1.43	78

The results of a two-way analysis of variance for repeated measures, shown in Table 14, reveal no statistically significant differences between groups in relation to levels of self-efficacy (*F*(1,76)=2.696, *p*>.05). Students in the control group reported slightly increased levels on posttest results while students in the LEGO group reported slightly lower levels compared to their pretest responses. The data do not support Hypothesis 5.

Table 14

Analysis of Variance Results for Group and Self-Efficacy

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	6.321	1	6.321	1.936	.168
Error 1	248.196	76	3.266		
			Within subjects		
SE	.016	1	.016	.041	.840
SE X Group	1.047	1	1.047	2.696	.105
Error 2	29.519	76	.388		

Hypothesis 6 states that students taking part in LEGO Mindstorms programming activities will show lower levels of test anxiety in relation to their programming course than students taking part in a traditional computer programming course. Table 15 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 15

Mean Levels of Test Anxiety

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	3.95	1.56	38	3.93	1.63	38
LEGO	3.87	1.28	40	3.77	1.57	40
Combined	3.91	1.42	78	3.85	1.59	78

While the levels of test anxiety did decrease more in the LEGO group than the control group, results from a two-way ANOVA for repeated measures shown in Table 16 reveal there was not a statistically significant difference between the groups in relation to levels of test anxiety ($F(1,76)=.159$, $p>.05$). The data do not support Hypothesis 6.

Table 16

Analysis of Variance Results for Group and Test Anxiety

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	.556	1	.556	.134	.715
Error 1	315.630	76	4.153		
Within subjects					
TA	.131	1	.131	.300	.585
TA X Group	.069	1	.069	.159	.691
Error 2	33.095	76	.435		

Summary

The results taken from student MSLQ surveys suggest that the use of LEGO Mindstorms programming activities had very little effect on student motivation. Tests revealed no statistically significant differences between groups in relation to intrinsic goal orientation, task value, control of learning beliefs, self-efficacy, and test anxiety at the alpha = .05 level. The experiment did detect a statistically significant difference between groups in relation to extrinsic goal orientation. The LEGO group, however, showed a larger decrease in levels of extrinsic goal orientation, suggesting they were less motivated in learning the material for rewards such as grades. This is not necessarily a bad thing, but ideally a decline in extrinsic goal orientation would be accompanied by a statistically significant increase in intrinsic goal orientation suggesting students were instead motivated because they were interested in the content of the material. This was not the case in this study.

LEGO Mindstorms Effect on Learning Strategies

The second research question of this study was "How will using LEGO Mindstorms programming activities affect the learning strategies of students in a university introductory computer programming course?" Hypotheses 7-15 were used to test different aspects of this area.

Hypothesis 7 states that students taking part in LEGO Mindstorms programming activities will use more rehearsal strategies than students taking a traditional programming course. Table 17 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 17

Mean Levels of Rehearsal

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	4.13	1.35	38	4.14	1.46	38
LEGO	3.96	1.26	40	4.38	1.44	40
Combined	4.04	1.30	78	4.26	1.44	78

The results of a two-way analysis of variance for repeated measures, shown in Table 18, reveal there was no statistically significant difference between groups in relation to levels of rehearsal (*F*(1,76)=2.388, *p*>.05). The control group reported similar levels of rehearsal strategies on both occasions while the LEGO group reported slightly lower levels of rehearsal strategies on the posttest. The data do not support Hypothesis 7.

Table 18

Analysis of Variance Results for Group and Rehearsal

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	.037	1	.037	.012	.914
Error 1	236.111	76	3.107		
			Within subjects		
Reh	1.763	1	1.763	2.543	.115
Reh X Group	1.655	1	1.655	2.388	.126
Error 2	52.680	76	.693		

Hypothesis 8 states that students taking part in LEGO Mindstorms programming activities will use more elaboration strategies than students taking a traditional programming course. Table 19 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 19

Mean Levels of Elaboration

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	4.32	1.17	38	4.38	1.20	38
LEGO	4.17	1.29	40	4.36	1.39	40
Combined	4.24	1.23	78	4.37	1.29	78

While posttest levels of elaboration went up in both groups, a two-way analysis of variance for repeated measures, reported in Table 20, showed there was no statistically significant difference between groups in relation to elaboration (*F*(1,76)=.386, *p*>.05). The data do not support Hypothesis 8.

Table 20

Analysis of Variance Results for Group and Elaboration

Source	*SS*	*df*	*MS*	*F*	*p*
		Between subjects			
Group	.262	1	.262	.095	.759
Error 1	209.714	76	2.759		
		Within subjects			
Elab	.644	1	.644	1.415	.238
Elab X Group	.176	1	.176	.386	.536
Error 2	34.591	76	.455		

Hypothesis 9 states that students taking part in LEGO Mindstorms programming activities will use more organizational strategies than students taking a traditional programming course. Table 21 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 21

Mean Levels of Organization

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	3.93	1.33	38	4.01	1.43	38
LEGO	3.82	1.54	40	3.86	1.41	40
Combined	3.88	1.43	78	3.93	1.41	78

There was a slight increase in the use of organizational skills by students in both groups in terms of their posttest results. A two-way ANOVA for repeated measures, however, determined there was no statistically significant result between groups in relation to the use of organizational strategies (*F*(1,76)=.020, *p*>.05). These results are shown in Table 22. The data do not support Hypothesis 9.

Table 22

Analysis of Variance Results for Group and Organization

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	.688	1	.688	.196	.659
Error 1	267.290	76	3.517		
			Within subjects		
Org	.118	1	.118	.203	.653
Org X Group	.012	1	.012	.020	.887
Error 2	43.966	76	.579		

Hypothesis 10 states that students taking part in LEGO Mindstorms programming activities will show higher levels of critical thinking strategies than students taking a traditional programming course. Table 23 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group. Both groups showed an increase in critical thinking strategies on posttest results compared to pretest levels. The control group, however, reported higher levels of critical thinking strategies than the LEGO group.

Table 23

Means Levels of Critical Thinking

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	3.94	1.38	38	4.12	1.40	38
LEGO	3.46	1.61	40	3.87	1.67	40
Combined	3.69	1.51	78	3.99	1.54	78

Table 24 shows the results from a two-way ANOVA for repeated measures. There was no statistically significant difference between groups in relation to levels of critical thinking strategies ($F(1,76)=1.021$, $p>.05$). The data do not support Hypothesis 10.

Table 24

Analysis of Variance Results for Group and Critical Thinking

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	5.305	1	5.305	1.290	.260
Error 1	312.468	76	4.111		
			Within subjects		
CT	3.320	1	3.320	6.232	.015
CT X Group	.544	1	.544	1.021	.315
Error 2	40.485	76	.533		

Hypothesis 11 states that students taking part in LEGO Mindstorms programming activities will show higher levels of metacognitive self-regulation strategies than students taking a traditional programming course. Table 25 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group. Levels of metacognition remained constant within the control group while there was an increase within the LEGO group between pretest and posttest occurrences. The LEGO group levels of metacognitive strategies were lower in comparison to control group levels.

Table 25

Mean Levels of Metacognitive Self Regulation

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	4.24	.85	38	4.23	1.05	38
LEGO	4.02	1.12	40	4.16	1.25	40
Combined	4.13	1.00	78	4.19	1.15	78

A two-way ANOVA for repeated measures, shown in Table 26, failed to show a statistically significant interaction effect between groups ($F(1,76)=.748$, $p>.05$). The data do not support Hypothesis 11.

Table 26

Analysis of Variance Results for Group and Metacognitive Self Regulation

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	.820	1	.820	.396	.531
Error 1	157.564	76	2.073		
			Within subjects		
MSR	.141	1	.141	.536	.466
MSR X Group	.197	1	.197	.748	.390
Error 2	20.004	76	.263		

Hypothesis 12 states that students taking part in LEGO Mindstorms programming activities will show better time and study environment strategies than students taking a traditional programming course. Table 27 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group. The self-reported control group results slightly decreased on posttest levels while the LEGO group slightly increased.

Table 27

Mean Levels of Time and Study Environment

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	4.40	.90	38	4.31	1.13	38
LEGO	4.32	1.29	40	4.39	1.24	40
Combined	4.36	1.11	78	4.35	1.18	78

A two-way ANOVA for repeated measures, however, failed to show a statistically significant interaction effect between groups (*F*(1,76)=.999, *p*>.05). These results are shown in Table 28. The data do not support Hypothesis 12.

Table 28

Analysis of Variance Results for Group and Time and Study Environment

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	.000	1	.000	.000	.998
Error 1	181.220	76	2.384		
			Within subjects		
TSE	.004	1	.004	.014	.905
TSE X Group	.283	1	.283	.999	.321
Error 2	21.519	76	.283		

Hypothesis 13 states that students taking part in LEGO Mindstorms programming activities will show better effort regulation strategies than students taking a traditional programming course. Table 29 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 29

Mean Levels of Effort Regulation

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	4.85	1.14	38	4.77	1.07	38
LEGO	4.51	1.55	40	4.48	1.48	40
Combined	4.68	1.37	78	4.62	1.30	78

Both students in the control and LEGO groups reported slightly lower levels of effort regulation on posttest results compared to pretest levels. A two-way ANOVA for repeated measures showed no statistically significant interaction effect between groups (*F*(1,76)=.039, *p*>.05). These results are shown in Table 30. The data do not support Hypothesis 13.

Table 30

Analysis of Variance Results for Group and Effort Regulation

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	3.802	1	3.802	1.277	.262
Error 1	226.327	76	2.978		
			Within subjects		
ER	.118	1	.118	.207	.650
ER X Group	.022	1	.022	.039	.844
Error 2	43.456	76	.572		

Hypothesis 14 states that students taking part in LEGO Mindstorms programming activities will show higher levels of peer learning strategies than students taking a traditional programming course. Table 31 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group. Levels of peer learning increased in both the control and LEGO groups on posttest results compared to pretest levels.

Table 31

Mean Levels of Peer Learning

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	3.49	1.54	38	3.73	1.53	38
LEGO	2.78	1.44	40	3.18	1.54	40
Combined	3.13	1.52	78	3.45	1.55	78

Results from a two-way ANOVA for repeated measures, shown in Table 32, reveal no statistically significant interaction effect between groups (*F*(1,76)=.428, *p*>.05). The data do not support Hypothesis 14.

Table 32

Analysis of Variance Results for Group and Peer Learning

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	15.288	1	15.288	3.852	.053
Error 1	301.672	76	3.969		
Within subjects					
PL	3.952	1	3.952	6.511	.013
PL X Group	.260	1	.260	.428	.515
Error 2	46.129	76	.607		

Hypothesis 15 states that students taking part in LEGO Mindstorms programming activities will show higher levels of help seeking strategies than students taking a traditional programming course. Table 33 shows means and standard deviations of pretest and posttest data from both the control and experimental LEGO group.

Table 33

Mean Levels of Help Seeking

	Pre Test			Post Test		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	3.95	1.50	38	3.91	1.32	38
LEGO	3.48	1.62	40	3.64	1.38	40
Combined	3.71	1.57	78	3.77	1.35	78

Students in the LEGO group reported increased levels of help seeking compared to their pretest levels. Students in the control group reported a slight decrease. A two-way ANOVA for repeated measures failed to show a statistically significant interaction effect between groups ($F(1,76)=.663$, $p>.05$). These results are shown in Table 34. The data in this study do not support Hypothesis 15.

Table 34

Analysis of Variance Results for Group and Help Seeking

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	5.196	1	5.196	1.420	.237
Error 1	278.039	76	3.658		
Within subjects					
HS	.147	1	.147	.246	.621
HS X Group	.397	1	.397	.663	.418
Error 2	45.567	76	.600		

Summary

Results taken from student MSLQ surveys failed to show any statistically significant differences between groups in relation to learning strategies. These different scales include rehearsal, elaboration, organization, critical thinking, metacognitive, time and study environment, effort regulation, peer learning, and help seeking strategies. Bergin et al. (2005) emphasized the importance of metacognitive and resource management in relation to student success in a computer programming course. It should be noted that students in the experimental LEGO group did show increases in posttest levels on all learning strategy scales except effort regulation. These increases were not found to be statistically significant at the alpha = .05 level when compared to the control group results.

LEGO Mindstorms Effect on Mastery of Course Objectives

The third and final major research question of this study was "How will using LEGO Mindstorms programming activities affect mastery of course objectives in a university introductory computer programming course?” This question focused on five hypotheses that were tested using exam scores. Exam two in the course served as a pretest for the groups in order to establish a starting reference point of their programming abilities. The final exam (exam three) was used to compare their final abilities after either receiving normal classroom instruction or instruction using LEGO Mindstorms. The results were consistent with Fagin and Merkle’s (2002) study, concluding that the use of LEGO Mindstorms robotic activities does not improve student ability in computer programming. Because the second and third exams were not identical, as in a true pretest / posttest experiment, raw exam percentages were converted into z-scores in order to obtain standardized data that could then be statistically compared. The result of each hypothesis is presented in the following section.

Hypothesis 16 states that students taking part in LEGO Mindstorms programming activities will perform significantly better on the course final exam than students taking a traditional programming course. Table 35 shows means and standard deviations of mastery percentages on exams two and three from both the control and experimental LEGO group. Students in both groups scored slightly lower on the final exam than they did on exam two. The two groups were also very similar in their scores when compared to each other.

Table 35

Means for Overall Exam Percentage Correct

	Exam 2			Exam 3		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	69.11	13.28	40	67.54	14.55	40
LEGO	68.75	13.74	43	66.07	13.14	43
Combined	68.93	13.44	83	66.78	13.77	83

A two-way analysis of variance for repeated measures using standardized z-scores revealed no statistically significant interaction effect between groups on exam scores (*F*(1,76)=.459, *p*>.05). These results can be seen in Table 36. The data do not support Hypothesis 16.

Table 36

Analysis of Variance Results for Group and Overall Exam Percentages

Source	*SS*	*df*	*MS*	*F*	*p*
			Between subjects		
Group	.183	1	.183	.097	.756
Error 1	152.164	81	1.879		
			Within subjects		
Exam	0	1	0	.001	.981
Exam X Group	.066	1	.066	.459	.500
Error 2	11.587	81	.143		

Hypothesis 17 states that students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of selection structures on the final course exam than students taking a traditional programming course. Table 37 shows means and standard deviations of selection structure percentages correct from both the control and experimental LEGO group. Students in the LEGO group scored higher on exam two than the control group but showed a larger decrease on final exam percentages. Both groups decreased in their selection structures mastery percentage on the final exam.

Table 37

Means for Percentage Correct on Selection Structures Objective

	Exam 2			Exam 3		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	66.25	13.10	40	62.08	24.46	40
LEGO	67.64	13.75	43	61.63	25.85	43
Combined	66.97	13.38	83	61.85	25.04	83

A two-way analysis of variance for repeated measures using standardized z-scores, seen in Table 38, revealed there was no statistically significant interaction effect between groups in relation to mastery of selection structures (*F*(1,76)=.308, *p*>.05). The data do not support Hypothesis 17.

Table 38

Analysis of Variance Results for Group and Selection Structures Percentage

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	.249	1	.249	.175	.677
Error 1	112.485	79	1.424		
Within subjects					
Exam	.001	1	.001	.004	.949
Exam X Group	.112	1	.112	.308	.580
Error 2	28.620	79.000	.362		

Hypothesis 18 states that students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of repetition structures on the final course exam than students taking a traditional programming course. Table 39 shows means and standard deviations of repetition structure percentage correct from both the control and experimental LEGO group. Although control group students showed a higher percentage of mastery on exam two, their final exam percentage was lower than the LEGO group.

Table 39

Means for Percentage Correct on Repetition Structures Objective

	Exam 2			Exam 3		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	63.64	17.47	40	57.08	29.21	40
LEGO	62.58	19.05	43	60.08	24.70	43
Combined	63.09	18.20	83	58.63	26.84	83

A two-way ANOVA for repeated measures, however, reveals that this difference was not statistically significant ($F(1,76)=.866$, $p>.05$). These results are shown in Table 40. The data in this study do not support Hypothesis 18.

Table 40

Analysis of Variance Results for Group and Repetition Structures Percentage

Source	*SS*	*df*	*MS*	*F*	*p*
Between subjects					
Group	.030	1	.030	.018	.895
Error 1	135.801	81	1.677		
Within subjects					
Exam	0	1	0	.001	.973
Exam X Group	.298	1	.298	.866	.355
Error 2	27.871	81	.344		

Hypothesis 19 states that students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of functions on the final course exam than students taking a traditional programming course. Table 41 shows means and standard deviations relating to mastery of the concept of functions. Both the control group and LEGO group showed decreases in the percentage of questions answered correctly on the final exam compared to exam two with the LEGO group showing a greater decrease in scores.

Table 41

Means for Percentage Correct on Functions Objective

	Exam 2			Exam 3		
	M	*SD*	*n*	*M*	*SD*	*n*
Control	77.88	14.28	40	73.82	13.82	40
LEGO	77.46	15.69	43	71.36	13.11	43
Combined	77.66	14.94	83	72.54	13.43	83

A two-way ANOVA for repeated measures using standardized z-scores, shown in Table 42, revealed that the difference was not statistically significant (*F*(1,76)=.390, *p*>.05). The data do not support Hypothesis 19.

Table 42

Analysis of Variance Results for Group and Functions Percentage

Source	*SS*	*df*	*MS*	*F*	*p*
		Between subjects			
Group	1.474	1	1.474	1.031	.313
Error 1	112.998	79	1.430		
		Within subjects			
Exam	.004	1	.004	.009	.923
Exam X Group	.168	1	.168	.390	.534
Error 2	33.984	79	.430		

Hypothesis 20 states that students taking part in LEGO Mindstorms programming activities will demonstrate better understanding of one-dimensional arrays on the final course exam than students taking a traditional programming course. Because the concept of one-dimensional arrays was not introduced in both groups until after exam two, this objective was not measured before the final exam. An initial comparison level was therefore unavailable. A comparison of percentage mastery on the final exam, however, can be done. These results may not be as clear due to the lack of an initial comparison. Table 43 shows means and standard deviations of one-dimensional array mastery from both the control and experimental LEGO group as observed on the final exam.

Table 43

Means for Percentage Correct on One-dimensional Arrays Objective

	Exam 3		
	M	*SD*	*n*
Control	62.78	19.86	40
LEGO	63.95	19.34	43
Combined	63.39	19.48	83

The LEGO group scored slightly higher than the control group on final exam questions relating to one-dimensional arrays. A one-way analysis of variance, however, shows that there was no statistically significant difference between groups. This data can be seen in Table 44. The data do not support Hypothesis 20.

Table 44

Analysis of Variance Results for Group and One-dimensional Arrays

Source	*SS*	*df*	*MS*	*F*	*p*
Between groups	.075	1	.075	.075	.785
Within group	81.925	81	1.011		
Total	82.000	82			

Summary

Results taken from exams two and three failed to show any statistically significant differences between groups in relation to mastery of course objectives. Students showed a decrease in percentage of questions correct on the final exam compared to exam two. This decrease was found to be consistent in both the control group and experimental LEGO group. A discussion of possible reasons for this decrease in student performance is contained in the next section.

CHAPTER 5

DISCUSSION AND CONCLUSIONS

The previous section detailed the results from the experimental study utilizing LEGO® Mindstorms® systems (LEGO Group, http://mindstorms.lego.com). The following chapter will provide a summary of these results and will discuss possible implications and explanations. In addition, a series of follow-up questions will be examined that will provide ideas for possible improvements in the future. Finally, a summary of conclusions and recommendations for future research will be provided.

Review of Research Questions and Results

Research Question 1 asked, "How will using LEGO Mindstorms programming activities affect the motivation of students in a university introductory computer programming course?" Figure 2 shows a summary of the posttest scores of each scale of the MSLQ relating to motivation.

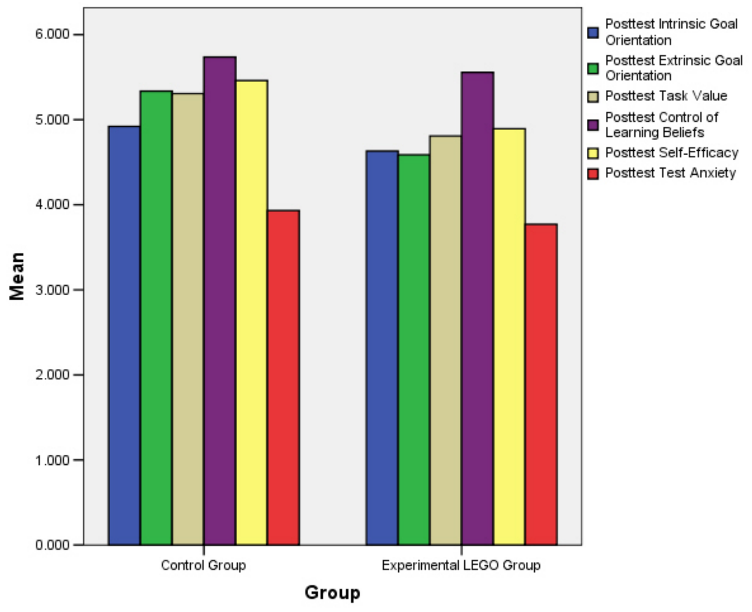


Figure 2. Bar chart showing summary of posttest MSLQ motivation results.

As explained in the previous chapter, only extrinsic goal orientation showed a statistically significant difference between groups when compared to pretest scores. To get a better understanding about the implications of these results, let us review the following findings of Bergin, Reilly, and Traynor (2005) based on their research study.

- The mean score of students with a high level of intrinsic goal orientation was found to be significantly higher than that of students with a medium or low level of intrinsic goal orientation. This supported their hypothesis that students with higher levels of intrinsic motivation will perform better in programming than students with lower levels of intrinsic motivation.
- No statistically significant difference was found between the mean scores of students reporting high levels of extrinsic motivation and those reporting medium levels of extrinsic motivation. Only one student reported low levels of extrinsic motivation and was therefore excluded in the comparison.
- Students with higher levels of intrinsic than extrinsic goal orientation were found to have a higher mean score than students with a higher level extrinsic than intrinsic goal orientation.
- Intrinsic goal orientation, task value, control of learning beliefs, and self-efficacy all had a statistically significant relationship with performance in the computer programming course.
- Text-anxiety and extrinsic goal orientation did not show statistically significant correlations with student performance.

The results of the LEGO Mindstorms experiment discussed in this dissertation suggest that while there were slightly negative results on some of the scales relating to student motivation, no statistically significant results could be found that would suggest that using LEGO Mindstorms programming activities to teach programming concepts in an introductory computer programming class either increases or decreases students' motivation. There was a significant decrease in extrinsic motivation after completing the LEGO activities, which suggests that the activities might lower the interest in rewards such as grades as a motivating factor. It should be noted, however, that the decrease in extrinsic motivation was not accompanied by an increase in intrinsic motivation, which stayed about the same. Still, the Bergin et al. (2005) study found that the combination of a higher level of intrinsic motivation and lower level of extrinsic motivation could lead to higher student performance in programming. Bergin's results, however, suggest that a decrease in extrinsic motivation alone might not be that meaningful in the overall motivation of students.

Research Question 2 asked, "How will using LEGO Mindstorms programming activities affect the learning strategies of students in a university introductory computer programming course?" Figure 3 summarizes the posttest results of both the control and experimental LEGO groups in relation to their learning strategies. Analysis failed to find any statistically significant differences between groups in relation to the use of learning strategies. The research of Bergin et al. (2005) examined the importance of these learning strategies in predicting success in an introductory computer programming course. The results of their study revealed the following.

- Students who reported higher levels of metacognitive and resource management strategies performed statistically significantly better than students reporting lower levels. Metacognitive strategies include critical thinking and metacognitive self-regulation (planning, monitoring, and regulating). Resource management strategies include time and study environment, effort regulation, peer learning, and help seeking.

- There was no statistically significant correlation between the use of cognitive learning strategies (rehearsal, elaboration, and organization) and student performance.

Bergin's results suggest that the use of metacognitive and resource management strategies are important factors in influencing student performance in an introductory computer programming course. The results of the LEGO Mindstorms study failed to show that the robotics activities had any statistically significant effects on the usage of any of the tested learning strategies.

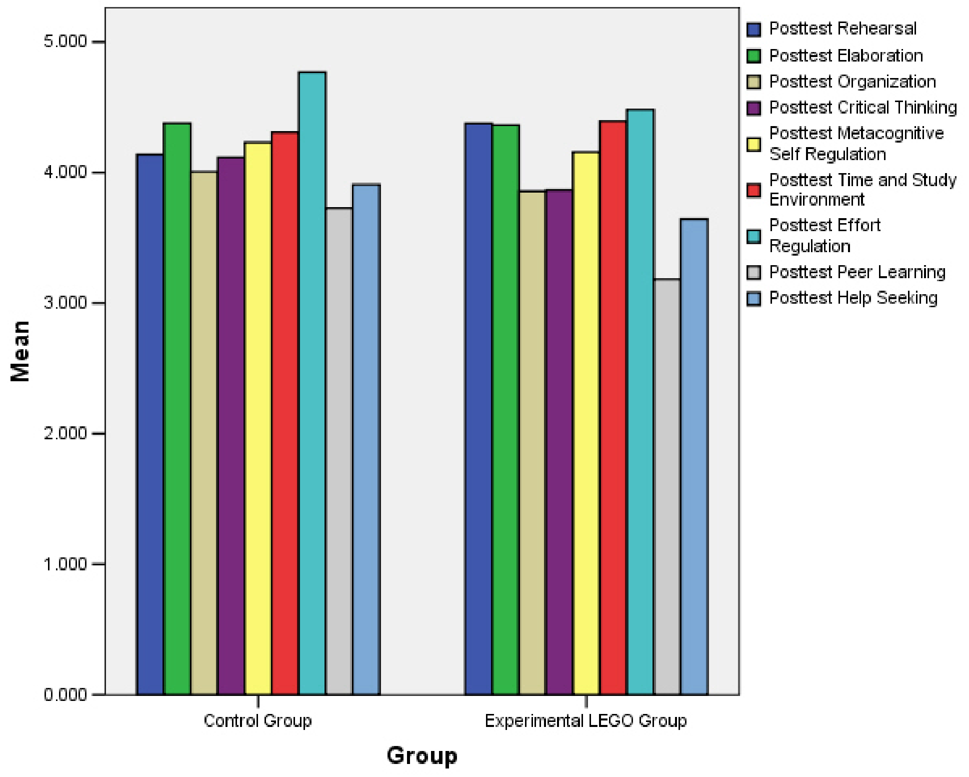


Figure 3: Bar chart showing summary of posttest MSLQ learning strategies results.

Research Question 3 asked, "How will using LEGO Mindstorms programming activities affect the mastery of course objectives in a university introductory computer programming course?" Figure 4 shows a summary of percentages correct on the final exam questions as well as individual course learning objectives.

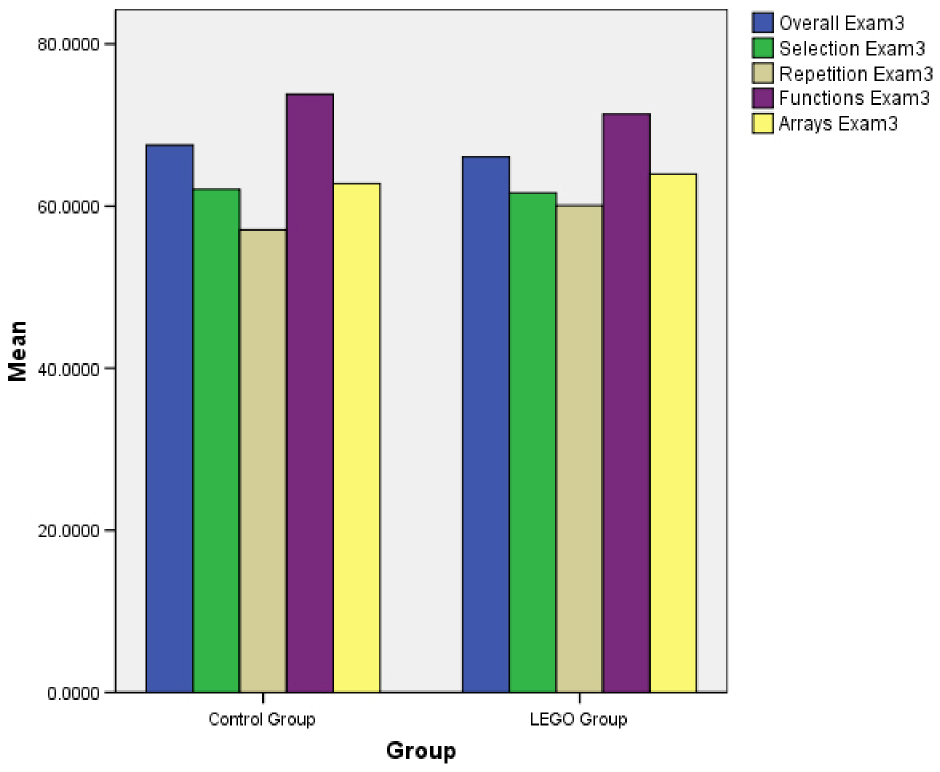


Figure 4: Bar chart showing final exam percentage results.

Analysis failed to reveal any statistically significant difference between groups in relation to exams and mastery of course learning objectives. This suggests that the use of LEGO Mindstorms robotics activities had little effect on the learning of computer programming concepts in the experimental group. These results were consistent with Fagin and Merkle's (2002) research at the United States Air Force Academy.

Discussion of Results

The results of this study suggest that the use of LEGO Mindstorms robotics activities had little if any effect in relation to student motivation, use of learning strategies, and mastery of course objectives in an introductory computer programming course. The next reasonable question to ask is "Why?" This section will focus on possible reasons for the lack of positive results.

SigCSE Panel Discussion

In March of 2006, a panel session discussion was conducted at the Computer Science Education Special Interest Group Conference in Houston, TX that posed the question, "Do LEGO Mindstorms Robots have a Future in CS Education?" The panel consisting of Myles McNally, Michael Goldweber, Barry Fagin, and Frank Klassner discussed the pros and cons of using LEGO Mindstorms to teach computer science. Klassner argued in support, Goldweber against, and Fagin provided a mixed opinion as well as discussing the results of his Air Force Academy research study. McNally served as the host and moderator for the discussion.

Goldweber argued that based on his department's use of Mindstorms-based robots he has concluded that their disadvantages outweigh their advantages and that these disadvantages can be categorized as both logistical and pedagogical. He mentioned increased cost as a major logistical disadvantage. Costs involved include not only that of the LEGO kits themselves, but also the cost of parts that break and have to be fixed or replaced. Due to the costs involved it is usually not possible to provide a LEGO robot for each student. Also, the use of the LEGO robot is usually limited to available lab time in class. Goldweber stated, "While this may be sufficient for students to complete their task-directed assignments, it is unlikely that this arrangement encourages or even supports assigned, or better yet, non-assigned, open ended, student-directed experimentation" (McNally et al., p. 61). The pedagogical disadvantages mentioned were mostly related to the limited ability to explore object-oriented concepts such as polymorphism. The introductory programming class at TAMU-Commerce used in this study has very little focus on object-oriented concepts, which are covered in the next programming course. One concept that had to be explained during the LEGO activities at TAMU-Commerce, however, was the difference between a function and a task. Because it is common for the LEGO robots to perform actions simultaneously, the concept of a task must be introduced. Tasks can be started and stopped and allow for the robot to do multiple things at the same time. For example, one task might instruct the robot to play a song. While it is playing the song another task running at the same time might be collecting light sensor readings. This is different from the concept of functions in that when a function is called, instructions in the calling function are paused until the instructions in the invoked function have been completed. The concept of tasks is not an invaluable one to teach to computer science students, but it could serve as a complication in an introductory programming course. Goldweber mentioned a few other disadvantages of Mindstorms robots including problems with consistency of robot movements due to differences in battery power, the need for frequent calibration of sensors, and problems associated with the engineering of the robots. These issues were encountered during LEGO lab sessions at TAMU-Commerce. The amount of battery charge caused consistency problems when writing turning and movement instructions. The amount of light in the lab had to be factored into any discussion of light sensor readings. Two LEGO motors were broken when the gears inside became stripped. Goldweber argued that such problems not only effect costs, but can also complicate programming lab exercises.

During the panel session, Fagin mentioned the opportunities that LEGO Mindstorms provide, but points out the lack of research to support the perceived benefits. “There is no evidence that LEGO Mindstorms improves learning, improves retention, attracts women to the major, reduces hair loss, cures acne, or any of the other miracle cures that those of us in computer science education are so desperate for" (p. 62). Fagin later stated that he believed the positives outweighed the negatives for using LEGO Mindstorms at the high school and middle school level. Although LEGO Mindstorms offer advantages such as being more kinesthetic, there are many limitations in their use. He pointed to the increased time it takes to receive feedback from changes in code compared to a normal programming environment. Changes are made to the source code, then must be compiled, then transferred to the robot, and then run. The limitation of the number of robots available and the normal inability to let students take them out of the lab was also mentioned. Fagin states that a software simulator could be used to address some of these limitations.

In addition to the limitations mentioned by Goldweber and Fagin, the lack of positive results relating to student motivation, use of learning strategies, and mastery of course objectives in this study have led to several questions.

- First, which students appear to be most in need of improvement regarding motivation and the use of learning strategies? A closer look at pretest scores based on gender, ethnicity, major, and classification might reveal some insight.
- Was the sample size large enough to detect differences between control and experimental groups?
- In an attempt to minimize possible feelings of jealousy, all the control group students were taken from the fall semester and all the LEGO students were taken from the spring semester. Do students tend to be more naturally motivated in one or the other?
- Due to time limitations, the assembly of the LEGO Mindstorms robots was done in an earlier semester outside of class. The experimental group did not have the opportunity to

actually build their robots. Did this have an effect on the motivational element of the activity?

- Were the LEGO robots perceived as too much of a child's toy to be effective for college students?
- Were the LEGO activities fun or did they serve as just something else to learn?
- Finally, as mentioned during the panel discussion, students were limited to the use of LEGO robots during class lab times and were not able to take them home. How did this affect the usefulness of the LEGO Mindstorms activities?

These questions were addressed through a combination of statistical analysis and a series of follow-up questions given to five students from the experimental LEGO group.

Demographic Factors

After examination of pretest results, statistically significant differences could only be found based on gender. Ethnicity, major, and classification did not appear to be important factors in relation to pretest differences in motivation, learning strategies, or mastery of course objectives. Figure 5 shows pretest levels of males and females based on their MSLQ results. As can be seen in Table 45, statistically significant differences could be found in relation to intrinsic goal orientation, task value, self-efficacy, test anxiety, and critical thinking at the alpha = .05 level. No significant gender based differences could be found in relation to course objectives.

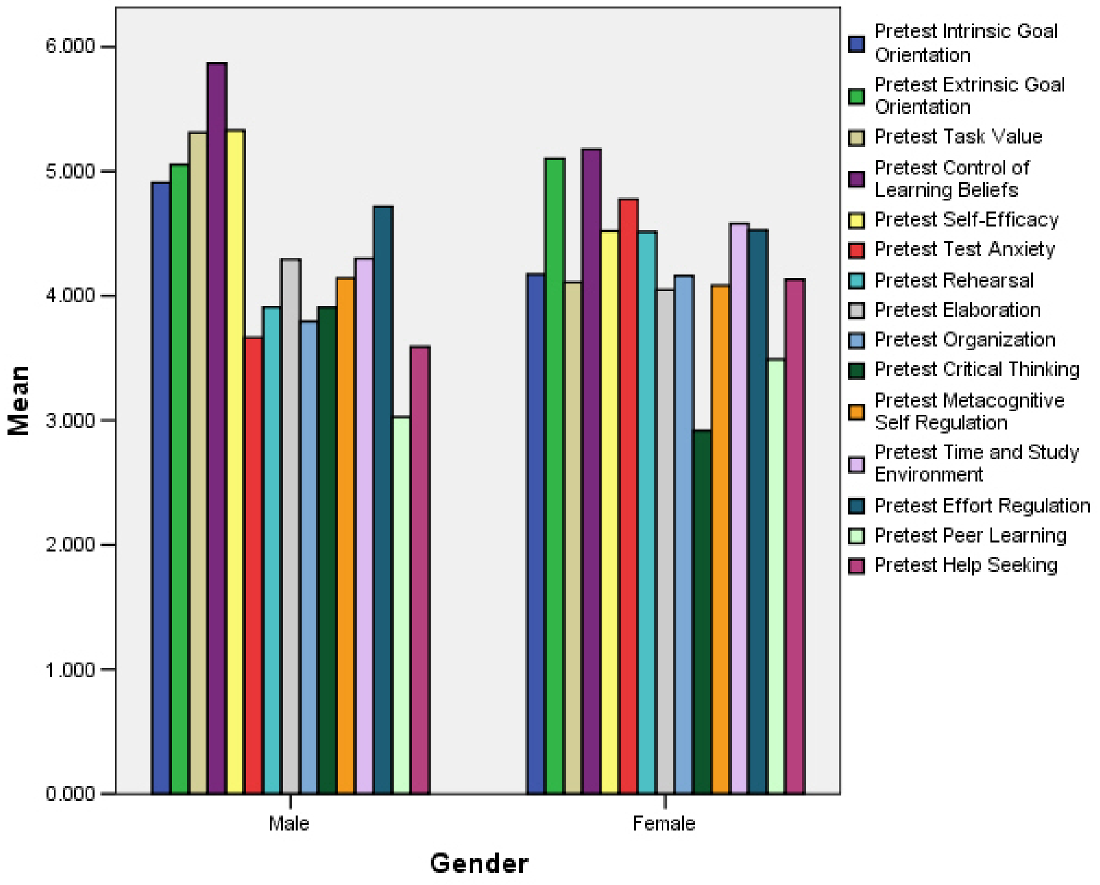


Figure 5: Bar chart showing pretest MSLQ gender results.

Table 45

Statistically Significant Pretest Analysis of Variance Results Based on Gender

	Male (*n* = 61)		Female (*n* = 17)		ANOVA	
Variable	*M*	*SD*	*M*	*SD*	*F*(1, 77)	*p*
Intrinsic Goal Orientation	4.91	1.21	4.17	1.53	4.39	.039
Task Value	5.31	1.46	4.11	1.64	8.60	.004
Self-Efficacy	5.33	1.24	4.52	1.30	5.55	.021
Test Anxiety	3.67	1.27	4.78	1.61	9.03	.004
Critical Thinking	3.91	1.51	2.92	1.27	6.07	.016

Even though the size of the gender groups was not equal, the apparent difference between males and females in computer science is definitely a topic worthy of future research endeavors. A number of studies have been done in this area. For example, Anderson, Welch, and Harris (1983) linked the low level of females in computer science courses to four social factors. These were parental encouragement directed toward sons rather than daughters, boy and girl peer groups widening the gap, stereotyped game software directed at boys, and lack of female role models both in the classroom and in the media.

A study conducted by Fisher, Margolis, and Miller (1997) also reported some interesting findings relating to gender differences in computer science. They found that although there was a gap in the level of previous experience and confidence with computer programming, female students exhibited equally strong ability and confidence as males by the time they became upperclassmen. The study did suggest that males and females have different motivations for becoming computer science majors. While male students interviewed cited mostly intrinsic interests in computing, females cited a bigger picture of what the field could provide in addition to intrinsic interests.

One area relating to gender differences that is of interest to the LEGO Mindstorms study is self-efficacy. Bandura (1986) discussed the importance of self-efficacy and its relationship to females in choosing to study computer science. She stated:

> People’s beliefs in their efficacy influence the choices they make, their aspirations, how much effort they mobilize in a given endeavor, how long they persevere in the face of difficulties and setbacks, whether their thought patterns are self-hindering or self-aiding,

the amount of stress they experience in coping with taxing environmental demands, and their vulnerability to depression. (p. 257)

The results of the LEGO Mindstorms study are consistent with other research in suggesting a gender difference in the area of self-efficacy. There is still much work to be done to address this and other factors that limit the percentage of the female population that choose computer science as their field of study.

Sample Size

One problem that occurs in many research studies is the issue of sample size. Due to the limited number of students taking introductory computer programming at Texas A&M University – Commerce, a large sample was not practical for this study. Small sample sizes can have a noticeable effect on the power of any research study. One way to judge this impact is by conducting a power analysis. According to Rudestam and Newton (1992), a power analysis lets the researcher know how many subjects are necessary in order to detect any effects due to the independent variables. The level of power lets the researcher know the probability of having a Type II error. A Type II error occurs when the researcher fails to reject the null hypothesis even though it is false. In this situation an effect exists but is not detected by the study. The smaller the effect, the larger the sample size needs to be in order to detect it. Because of this possibility in the LEGO Mindstorms study, further research with a larger sample should be conducted.

Follow-up Interviews

In an attempt to shed some light on factors that may have influenced the LEGO Mindstorms study, a short series of follow-up questions were posed to a small, strategically selected group of five students. All of these students were members of the experimental LEGO group. Appendix A contains information showing each of the five student's results on pretest and posttest Motivated Strategies for Learning Questionnaire as well as course objective mastery percentages. Table 46 provides a demographic summary.

Table 46

Summary of Follow-Up Student Demographic Information

	Student 1	Student 2	Student 3	Student 4	Student 5
Gender	Male	Female	Male	Male	Female
Ethnicity	Caucasian	Caucasian	Caucasian	African American	Caucasian
Classification	Sophomore	Freshman	Post Bachelors	Sophomore	Junior
Major	Computer Science	Mathematics	B.S. in Math Completed	Industrial Engineering	Mathematics
Age	20 years old	18 years old	62 years old	19 years old	21 years old

Most of the questions asked of this group of students dealt with motivational aspects of the LEGO Mindstorms activities. The student responses suggest that overall the activities were enjoyable. Tables 47 and 48 show student responses to two questions regarding the LEGO activities in general.

Table 47

Follow-Up Responses Relating to Favorite Part of Course

Question	What was your favorite or most memorable part of your introductory programming class and why?
Student 1	My favorite part of csci151 happened to be the LEGO Mindstorms. I had used a similar program in 9th grade for a computer project, and really enjoyed the concept of controlling something robotic. It was really interesting to be able to give it another shot, this time with more knowledge and experience as a programmer. However, the most memorable thing was something my partner and I did with the Mindstorms project. We wanted to make it original, so we figured out how to make it play back sounds, and then programmed a song for it to play. When it reached the end of its list of programmed movements, it would then play the death march before coming to a stop. As a programmer, I've enjoyed finding cool tricks and treats to add to my programs, and that was by far one of the best.
Student 2	The robotics activities because it was more involved and hands on.
Student 3	I remember the LEGO activities at the end of the semester and being apprehensive because there had already been so much to learn in the course. Now we were starting something new in a different programming language. Even though the Not Quite C language was similar to C++, it was different enough to require learning some new syntax.

Student 4	I enjoyed seeing the power that my code could have on a physical object like the LEGO robot.
Student 5	I guess the LEGO activity.

Table 48

Follow-Up Responses Relating to LEGO Mindstorms Motivation Effectiveness

Question	Do you think the LEGO activities helped motivate students to be interested in computer programming?
Student 1	Honestly, I have no idea if it motivates students to be interested in computer programming because I was already interested in it. However, I feel it increased that interest because of my interest in robotics. It seems like a great tool for people who enjoy the hands on experience, but might not have been so for the people that were just getting by with the basics of C++.
Student 2	Yes, the students seem to be enjoying it.
Student 3	At first when we were going to do the LEGO activities, I was a little apprehensive about having to learn a new language, especially at that point in the semester. It would have been better to have introduced the LEGO stuff earlier in the semester and in the same language as the main course material. I would like to have had the LEGO activities spread out

	throughout the semester.
Student 4	Yes. It gave students the ability to have the freedom to control something real.
Student 5	Very much so. In most classes it is hard to be motivated to learn the material because most students do not see its use or purpose. By using the LEGO activities is class students are able to apply the knowledge and more than likely it will stay with the student longer.

One concern regarding the results of the LEGO Mindstorms study was the semester in which each group participated. All the students from the control group took the course in the fall and the LEGO group in the spring. This was done for convenience and to try to minimize any feelings of jealousy that the control group might have towards the LEGO group. The question that remains is whether or not students tend to be more naturally motivated in one particular semester. Responses from a follow-up question seen in Table 49 suggest that students have mixed feelings regarding which semester they think students tend to be more motivated.

Table 49

Follow-Up Responses Relating to Motivation and Semester

Question	Do you think students tend to be less motivated in the spring semester compared to the fall?
Student 1	Honestly, I tend to be less motivated in the fall than the spring. This is

	because the spring seems to go by so much faster, and it seems like everything is crammed into shorter periods, so I have to be on top of my game. In the fall, I would have just gotten back from a 2-3 month break and the longer the break, the longer it takes to get back on track again.
Student 2	I tend to be more motivated in the fall because in the spring I’m more ready for summer and don’t care as much about school.
Student 3	It would have been harder to do the LEGO activities in the fall because of the Thanksgiving burn out factor and then having to learn something new towards the end of the semester.
Student 4	I think students tend to be more motivated in the spring because they come into the semester with high hopes. If they have bad grades in the fall they enter the spring wanting to do better.
Student 5	Probably. Although the winter break is a whole month off, students are normally still burnt out during the spring semester. For me, the spring semester is the hardest. Especially after spring break.

One limitation of the LEGO Mindstorms experiment was that the limited amount of class time did not allow students the opportunity to build the actual LEGO robots. Each robot was assembled during a previous semester by student volunteers during outside class time. Most of the follow-up students indicated that it would have been enjoyable to have been able to build the robots.

Table 50

Follow-Up Responses Relating to LEGO Robot Building

Question	Would you have enjoyed the LEGO Mindstorms activities more if you had been able to build the actual robots?
Student 1	I think that I would have enjoyed that even more because then the robot would be even more custom. In my 9th grade class when I worked on the robotics project, we had to build our own design, and then program it to perform certain tasks. We created a device that would pick up the bed of a person in the ER and switch them over lightly and carefully to the next bed, without the need for doctors to pick up the sheet and move them, risking further injury, etc. Granted the project was scaled down, it was still interesting trying to overcome certain obstacles. But, if we were able to create our own custom LEGO Mindstorms robot here at college, then we would have needed a little bit more time to do that. It might even make a fun course: building and designing your own robot to perform certain tasks.
Student 2	Yes, that probably would have been more fun and interesting.
Student 3	Perhaps, but since it would have taken even more time possibly outside of class I would not have been very interested in doing it. I already had so many other things to do.
Student 4	Me personally, yes, because I'm an Industrial Engineering major but overall no, because some of the Computer Information Systems and Computer Science majors might not be into architecture and engineering

	stuff as much.
Student 5	Not sure. It would have been interesting to see how everything works, but it also probably would have been overwhelming to have had to build it and then program it for one assignment.

Even though LEGO Mindstorms are designed for ages twelve and up, the follow-up students in general didn't seem to simply think of them as toys. Their responses do suggest appreciation of their potential to teach programming concepts to middle and high school aged children.

Table 51

Follow-Up Responses Relating to Feelings about LEGO Mindstorms Activities

Question	What were your feelings about the LEGO activities? Did it seem too much of a child's toy / activity? Was it fun or just something else to learn?
Student 1	I really enjoyed the LEGO Mindstorms activities. Granted, at the beginning, I was lost, and it felt like it was one more thing to learn that I would never use again, but I eventually warmed up to it when I saw some of the possibilities. Eventually I saw it as a challenge and I for one love challenges once I see the possibilities. And I wouldn't have considered it a child's toy because it really felt like a physics lab. We were using light sensors, pressure sensors and a little math.
Student 2	A mix of both. It got us more involved. We did have more freedom to

	make the robot do what we wanted it to do.
Student 3	I was really impressed with it as a possible way to teach younger students, perhaps around twelve years old. You could use it as a way to teach some source code, so that they would have a greater appreciation for programming and maybe spark their interest. For the college class though, it would have been better to spread the activities throughout the semester instead of just sticking it all towards the end of the semester.
Student 4	It was fun to me. I liked the ability to have the robots play music.
Student 5	I really enjoyed this activity. It was a fun application of the material. I did not think it was childish but a concrete example of the material.

Another limitation of using LEGO Mindstorms in the classroom is the risk of losing equipment if students are allowed to take the robots home. Because the same robots had to be used for multiple course sections, students were required to complete all LEGO lab assignments during class time. Student follow-up questions as seen in Table 52 confirm that this was a weakness.

Table 52

Follow-Up Responses Relating to LEGO Robot Availability

Question	Did the lack of outside class availability affect the effectiveness of the LEGO activities?
Student 1	I guess that would depend on the person and their interest in the topic. My

	lab partner and I were looking up things we could do extra outside of class time, so that when we got to work on it in the lab, we were ready to implement some of those ideas.
Student 2	Yes, especially because of the limited amount of time we had in class.
Student 3	Yes, I'm used to working by myself and the limitations of working in partners did not allow me to fully understand all that we were doing.
Student 4	Yes, but it is better to be safe and keep control of the robots by not letting them out of the lab.
Student 5	For me, no. There was not a time that I needed time outside of class to work on it. It would have been frustrating though if I needed to spend time outside of class to work on it if that was not an option.

When asked about the enjoyableness of the activities in relation to gender, follow-up answers were mixed but hint that the LEGO activities, while still fun for some females, might be considered more of a male oriented activity.

Table 53

Follow-Up Responses Relating to LEGO Mindstorms and Gender

Question	Do you think LEGO Mindstorms activities are equally motivating for guys and girls?
Student 1	I guess that would depend on whether the girl has a thing for technological

	“toys” and gizmos like most guys tend to have an inherent interest in.
Student 2	From my point of view, no, but I could see how it would be for others. I thought it was fun, though.
Student 3	From my observations in the class it appeared that both guys and girls seemed equally motivated. The students that didn’t seem interested, it wasn’t based on gender. For some students, it seemed to actually hurt their understanding of the course material because it was more to learn and understand.
Student 4	I think it was more motivating for the guys due to the mechanical nature of the robots.
Student 5	Not sure.

Results from the MSLQ suggested that LEGO Mindstorms activities did little to influence students’ use of learning strategies in their computer programming course. Student follow-up questions confirm this. Responses can be seen in Table 54.

Table 54

Follow-Up Responses Relating to LEGO Mindstorms and Learning Strategies

Question	Did the LEGO Mindstorms activities have any influence on your learning and study strategies in your introductory programming class?
Student 1	No, not really.

Student 2	Perhaps some in terms of learning because it was more hands on.
Student 3	No, not really but it was probably because it was put in at the end of the semester.
Student 4	No, not really because it required me to put in extra work. It did add to my knowledge of programming, though.
Student 5	Not really. It may have helped learn the concepts better but it did not affect the way I studied.

When asked about whether or not the LEGO Mindstorms activities helped in their understanding of computer programming concepts, follow-up responses were mixed. The main factor that seemed to influence responses was the amount of previous experience with programming.

Table 55

Follow-Up Responses Relating to LEGO Mindstorms and Programming Concepts

Question	Did the LEGO Mindstorms activities help in your understanding of computer programming concepts such as selection structures, loops, functions, and arrays?
Student 1	Once again, it didn't. I already had a pretty good background with loops, functions, and arrays at that point. However, I think it would prove a good tool for reinforcing that knowledge if you have at least a basic understanding of those tools.

Student 2	Yes, it showed us more examples and you could see the effects of your programs on a real object instead of it being more like working on math equations.
Student 3	In my case no, but I was already familiar with those concepts. It was the first time I had seen my code affect a physical object.
Student 4	Yes, it helped me better understand programming. Industrial Engineering has such a wide range of areas in the field. It helped me learn more about programming but not necessarily other skills relating to my major.
Student 5	Yes, I had never had any experience with computer science before so learning some of the concepts was difficult without concrete examples. Using this activity was a great way to provide an example for students in which they can understand the use and purpose of different programming concepts.

Finally, when asked about possible improvements that could be made to make the LEGO Mindstorms activities more effective, most of the follow-up students stated that more time or time spread out throughout the semester with the robots would have been better.

Table 56

Follow-Up Responses Relating to Possible LEGO Mindstorms Improvements

Question	What improvements can you think of that would make the LEGO Mindstorms programming activities more motivating and effective for

	student learning of computer programming concepts?
Student 1	I think that the only thing that would make it more effective would be more time to work with the LEGO Mindstorms. It seems that the short period of time we had to work with LEGO Mindstorms was just enough to get the taste in our mouth, but not enough for us to really figure out what all we had in front of us.
Student 2	Perhaps if we could do more with the robots so we could see more kinds of things.
Student 3	Integrating the activities throughout the semester if there was a way of doing the activities without having to lose a lot of class time.
Student 4	I enjoyed the LEGO Mindstorms activities but I noticed that some of the students didn't seem to be that interested in it. I think some kind of activities that related more to what students have going on in their life would be more effective.
Student 5	Can't think of any.

Project STEEM

Although the LEGO Mindstorms study described in this dissertation failed to produce strong positive results, it should be pointed out that this does not mean that the use of LEGO Mindstorms is not effective. The robotics activities in the study lasted for only three weeks within the last five weeks of the semester. Spreading out the activities during the semester may have been more beneficial. Also, perhaps other alternative programming activities would prove to be more enjoyable for college students. Even Seymour Papert's (1980) work with LOGO was done with younger students.

I had the opportunity to work with groups of middle and high school students in a recent summer workshop program. The grant funded program was aimed at exposing secondary students to areas of science, technology, engineering, and mathematics in the hopes that they would pursue related interests in college. The students were split up into three different age groups with about twenty students in each group. I worked with each group using the newer LEGO Mindstorms NXT systems. This updated version of LEGO Mindstorms comes with a more powerful microprocessor, a light sensor, motors that can also sense rotation, an ultrasonic sensor that can detect distances to objects, and a sound sensor. There was an introductory session describing the system and activities, a construction session in which the students built the robots, and a programming session in which each group experimented with programming their LEGO robot using the drag and drop LEGO programming interface included in the kit. Working with three distinct age groups allowed me to make unstructured observations about the activities.

During the building sessions, students in the middle school group seemed to be the most engaged. All the students in the group seemed to work well together and were all involved. The oldest group comprised of high school juniors and seniors overall were not as engaged in the building process. Most of the students in each group were involved but there were a few just sitting around watching. There also seemed to be a difference between the boys and girls in the older group. The boys seemed to be more into the building than girls overall. The third group was comprised of freshman and sophomore high school students. This group seemed to have a good balance of involvement amongst boys and girls.

The biggest differences I observed during the programming sessions dealt with the amount of student involvement and the need for help. Students in the middle school group were very engaged in the programming activities but required more help in figuring things out. They were very determined to get things to work however, and it was very apparent that they were building their problem solving skills. The oldest group consisting of juniors and seniors needed much less assistance but not all students were participating equally in the activities. Some were sitting around and letting one or two others do all the work. Also, some of the older students seemed to move at a much faster pace and some slower. As with the building sessions, students in the freshman and sophomore group seemed to all be participating equally.

These observations, although very preliminary, seem to suggest that perhaps LEGO Mindstorms activities are best suited for students in middle and early high school. This would be an excellent opportunity for additional research.

Conclusions and Suggestions for Future Research

The research study described in this dissertation examined the possible link between self-regulated learning and LEGO Mindstorms robotic activities in teaching computer programming concepts in an introductory university computer programming course. The areas of motivation, learning strategies, and mastery of course objectives were investigated. In all three cases, statistical analysis failed to reveal any significant differences between the traditional control group and the experimental LEGO Mindstorms group as measured by the Motivated Strategies for Learning Questionnaire and course exams. Possible reasons for the lack of positive results include technical problems and limitations of the LEGO Mindstorms systems, limited number and availability of the robots outside of class, limited amount of time during the semester for the robotic activities, and a possible difference in effectiveness based on gender. The relatively small sample size in the study could also have limited the ability to detect differences between groups. It should be noted that the lack of positive results does not mean that the use of LEGO Mindstorms robotics activities to teach computer programming is not worth pursuing. In fact, responses to student follow-up questions suggest that at least some of the students really enjoyed the LEGO activities. As with any teaching tool or activity, there are numerous ways in which LEGO Mindstorms can be incorporated into learning. This study utilized LEGO Mindstorms robotic activities in a university introductory computer programming course by supplementing course material with a three week long series of robotics labs. This approach was chosen both for reasons relating to experimental design and because computer science education literature revealed the practice as a common way of incorporating LEGO Mindstorms into introductory computer programming courses. Some of the responses to follow-up questions in the study suggest that spreading out the robotic activities throughout the entire semester might be more effective and less intimidating. This approach would also allow the group of students who lose interest in the course early on in the semester to participate in the LEGO Mindstorms activities. It is also possible that while LEGO Mindstorms activities seem well suited for middle and high school students, they might not be the best choice at the university level. Perhaps other activities that are more in line with the interests of college aged students would be more effective in motivating them to learn computer programming. The challenge is that college students are very different from each other in terms of interests and abilities. The truth may be that no one particular teaching technique can appeal to the motivations of all students. Barry Fagin (McNally et al., 2006) stated, “there is no magic bullet that completely addresses the unique needs of our students in computer science education” (p. 62). He argued that computer science teachers should explore all available techniques before investing solely in LEGO Mindstorms. Perhaps there are other tools and learning environments that would be more effective and less expensive. Perhaps a combination or a choice of various activities would better appeal to students. The literature review of this dissertation discussed just a few of the numerous tools and environments that have been developed for this purpose. The effectiveness of some of these approaches has been documented in computer science journal articles. The problem of comparing them is due to the lack of consistency in terms of assessment. Gross and Powers (2005) discussed the issue of evaluating the assessment of novice programming environments. They introduced a rubric that could help in making comparisons and demonstrated their ideas through a survey of journal assessments. A formal study comparing a variety of these novice programming environments would be an excellent area for future research.

The results of this study suggest that technology and teaching environments can limit the effectiveness of even innovative learning activities, such as LEGO Mindstorms. Seymour Papert (1980s) argued that the computer should be considered a tool that allows students to be creative, in the same way a pencil is simply a tool that allows a poet to write poetry. In a speech to Japanese educators during the 1980s Papert stated,

> One of the things that’s wrong with school is that what you learn there, you can’t really use. Another thing that’s wrong with school is that there’s one way to do it. And that doesn’t happen in the real world either. In the real world, there are many ways to do things, and this is how creativity develops. This is how people make exciting new discoveries.

APPENDIX A

SUMMARY OF EXPERIMENTAL RESULTS FOR FOLLOW-UP STUDENTS

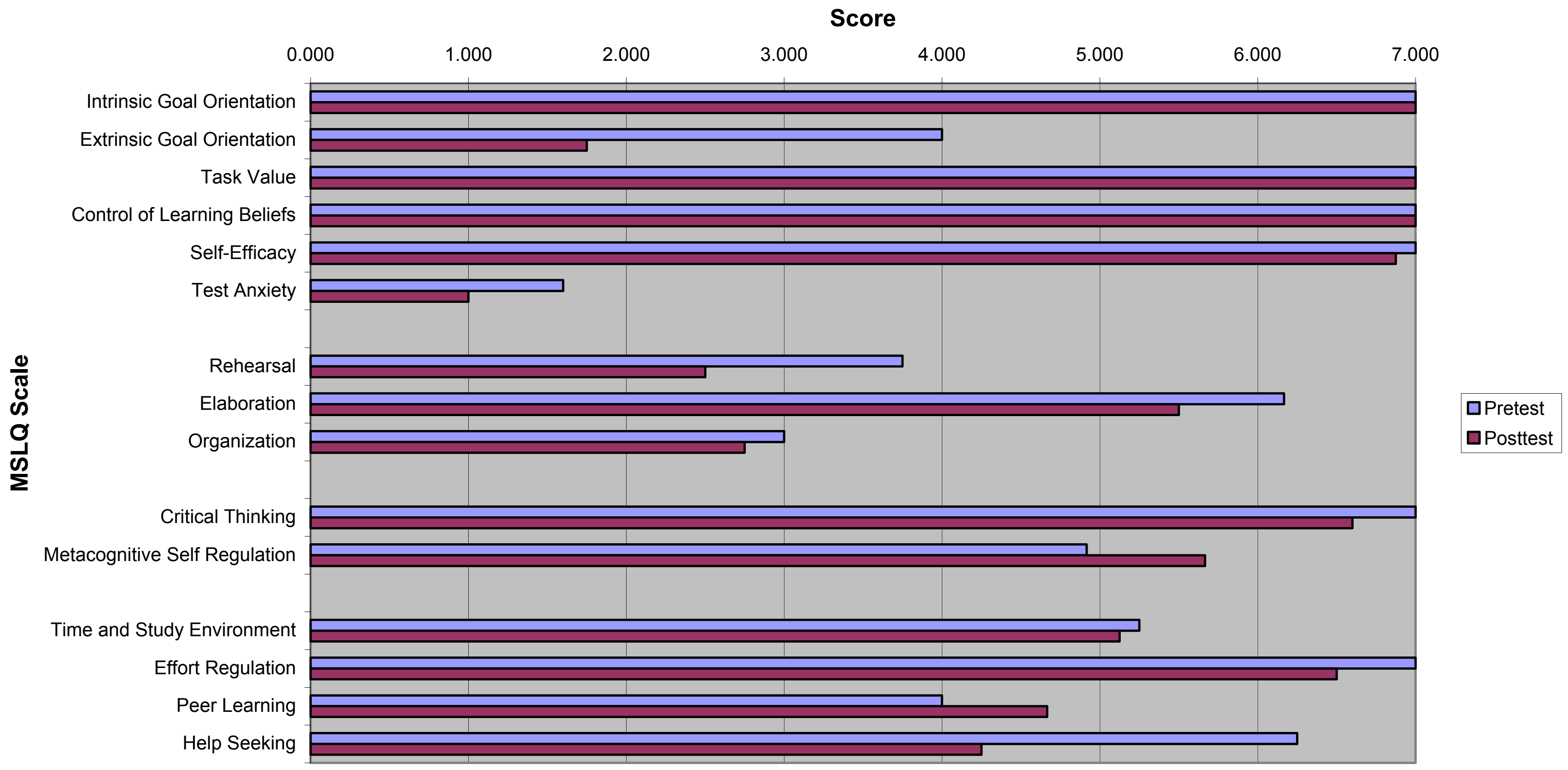


Figure 6: Bar chart showing MSLQ results for Student 1.

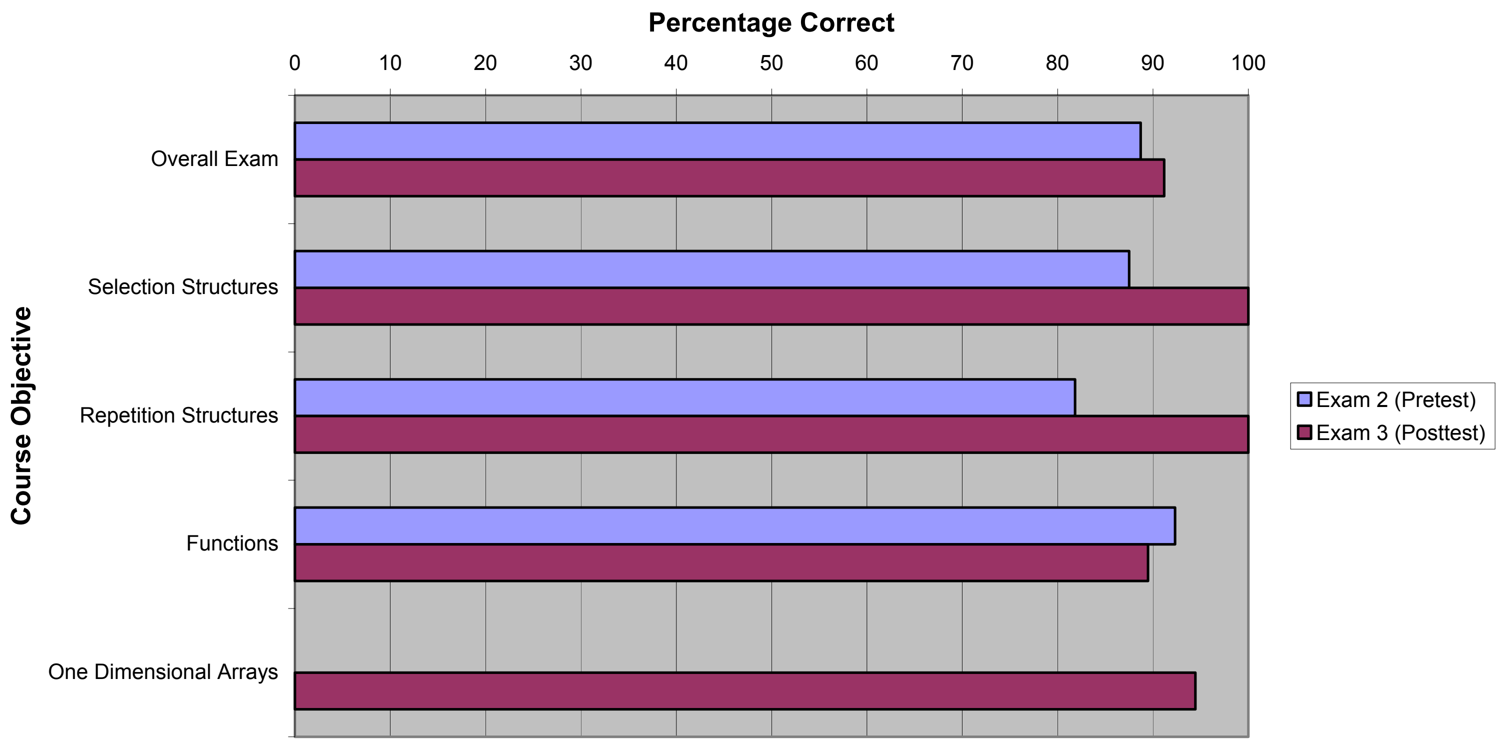


Figure 7: Bar chart showing mastery of course objectives for Student 1.

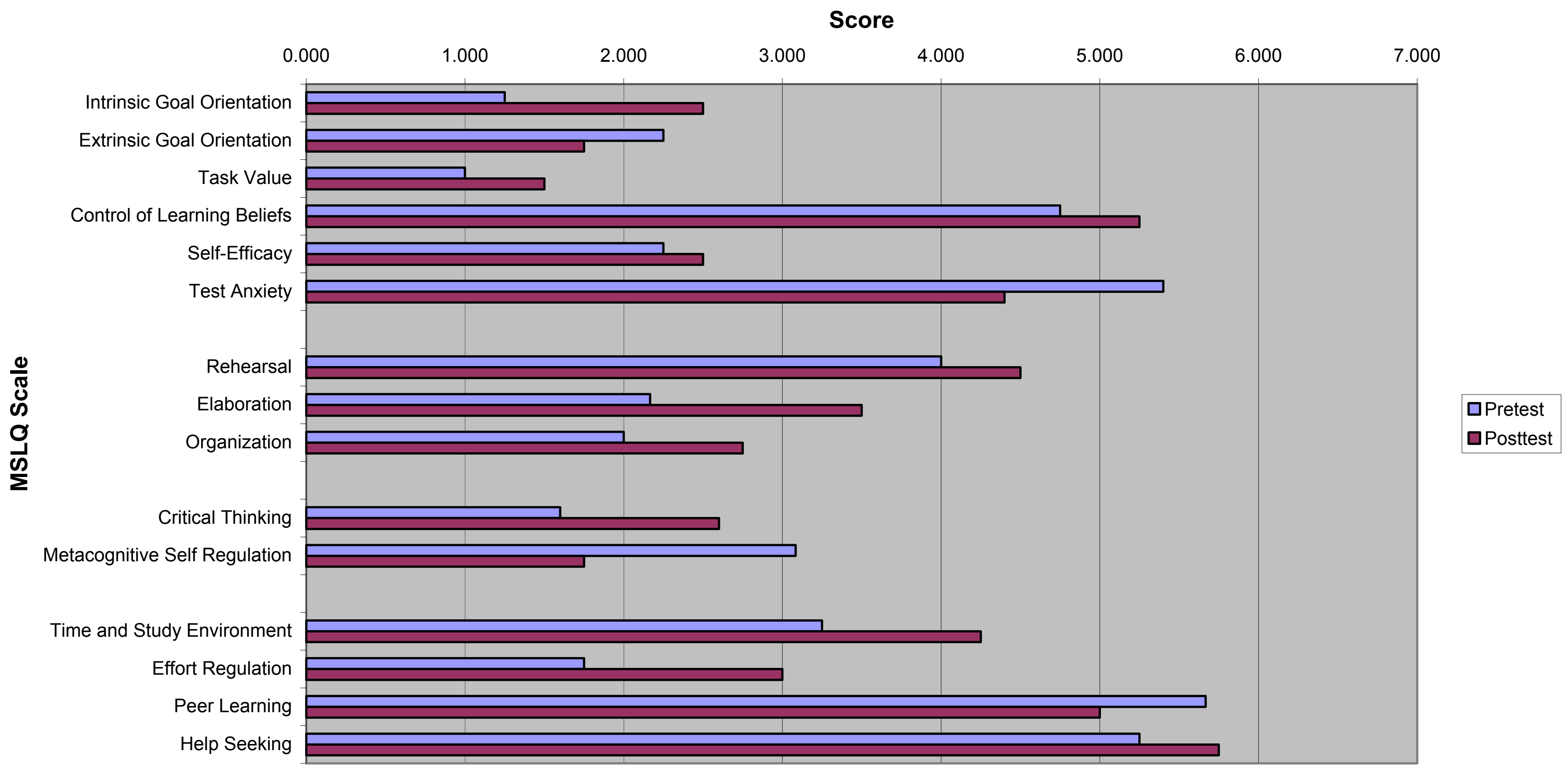


Figure 8: Bar chart showing MSLQ results for Student 2.

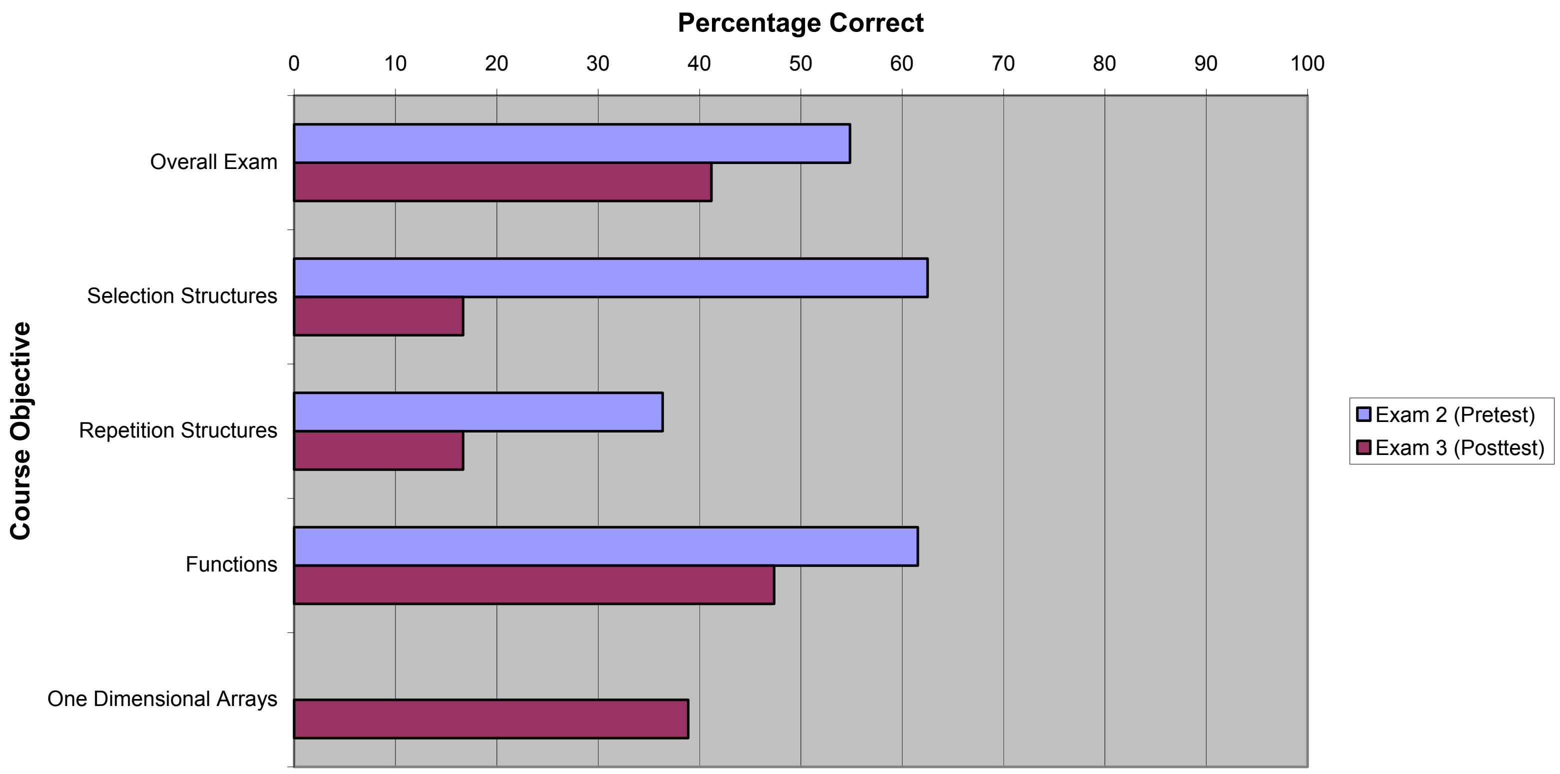


Figure 9: Bar chart showing mastery of course objectives for Student 2.

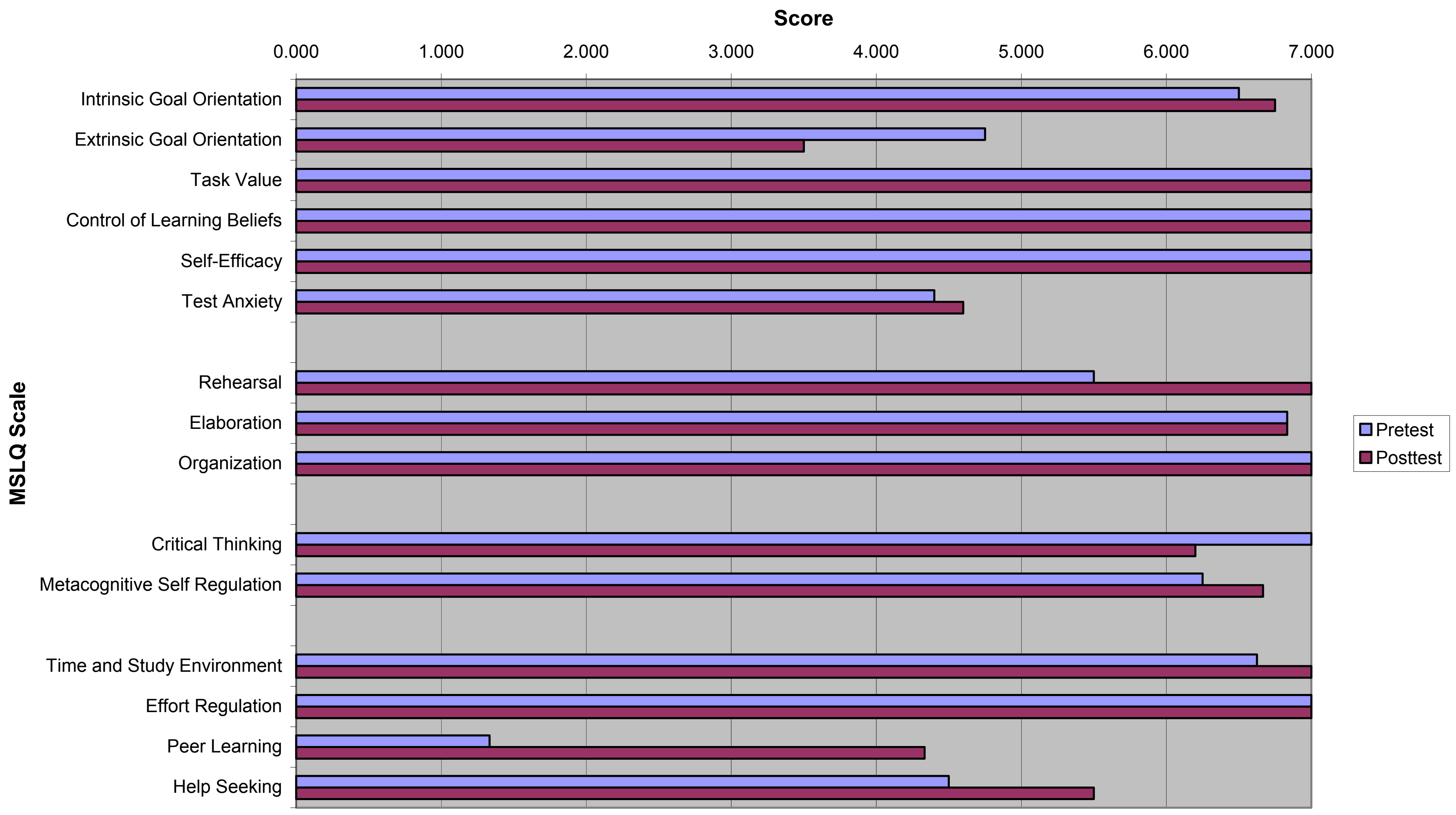


Figure 10: Bar chart showing MSLQ results for Student 3.

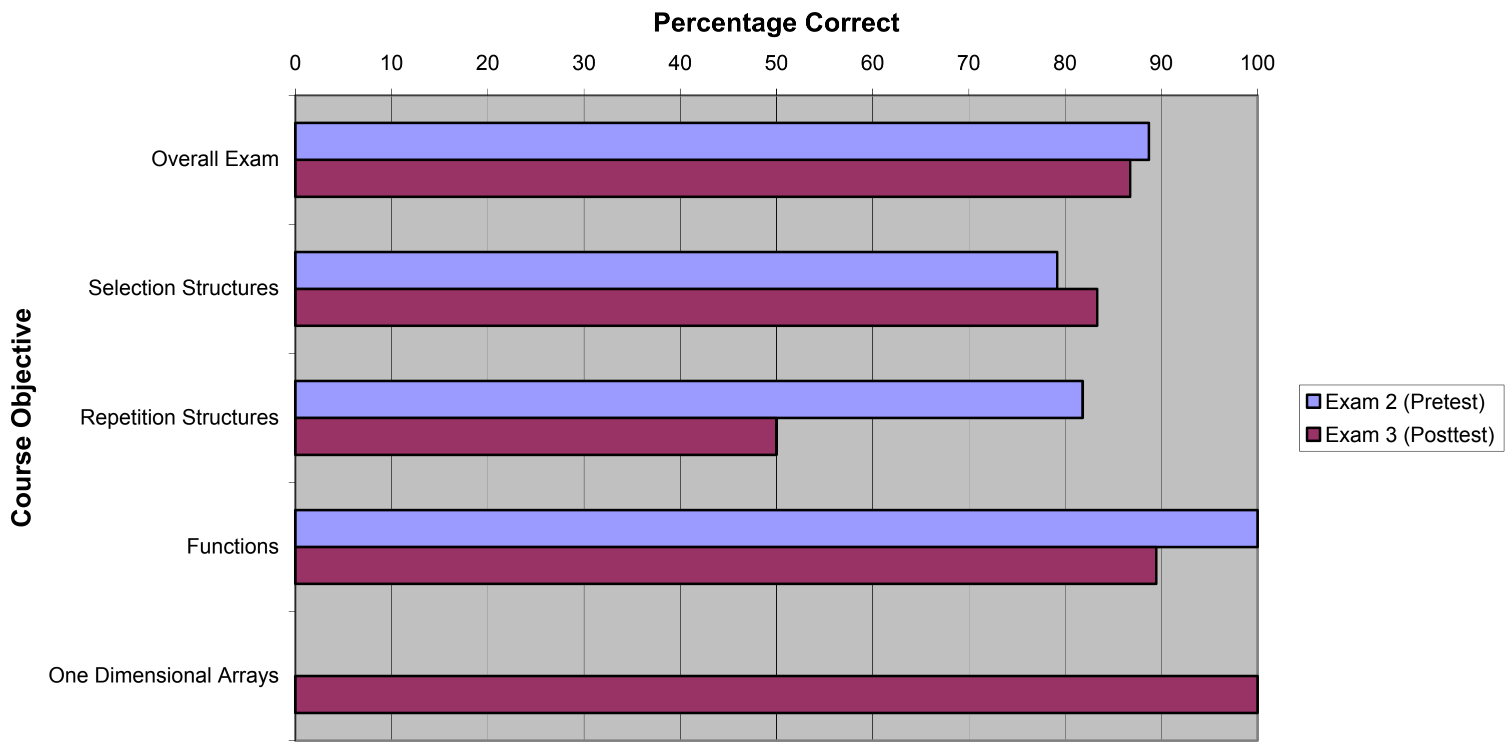


Figure 11: Bar chart showing mastery of course objectives for Student 3.

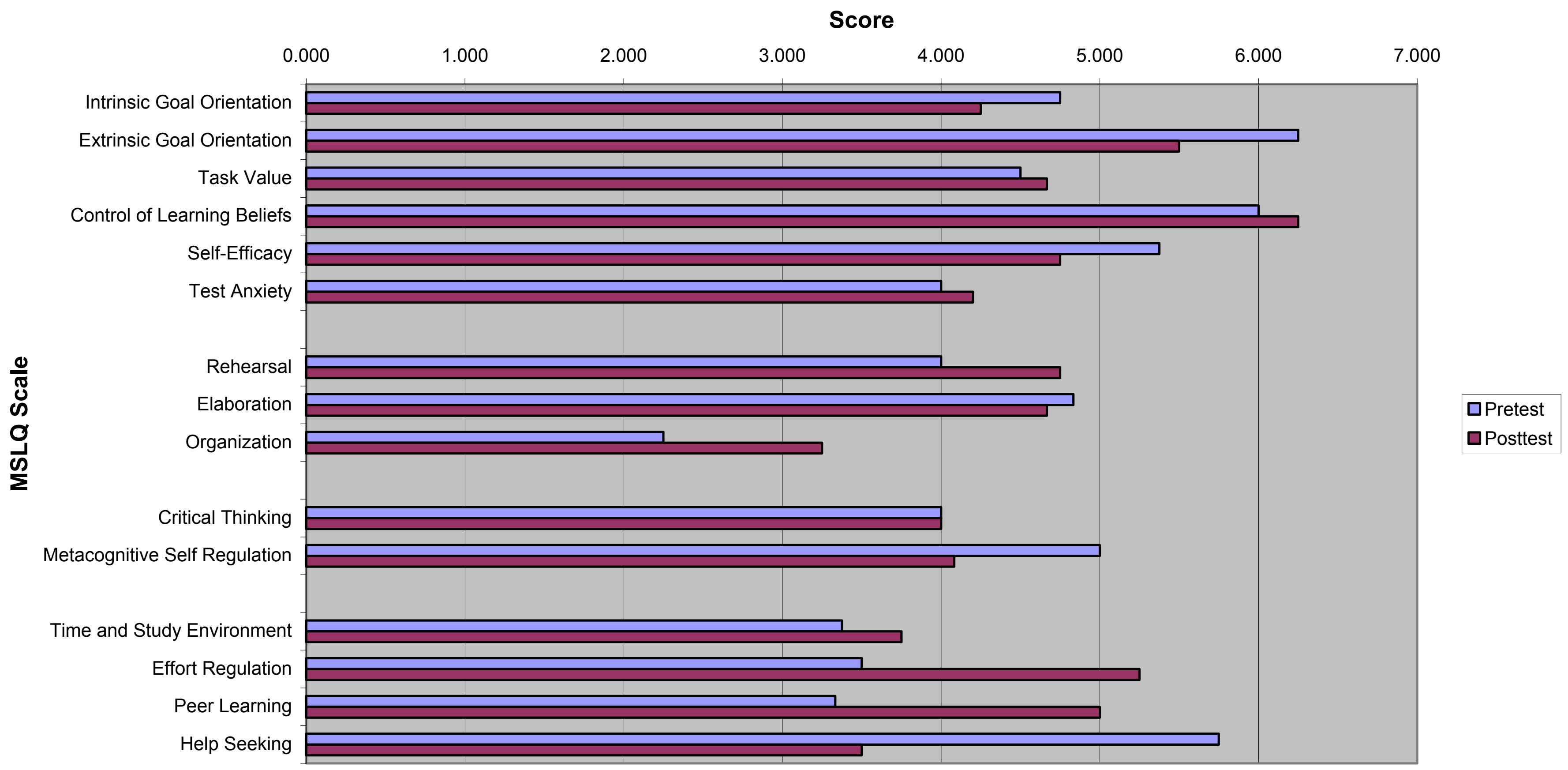


Figure 12: Bar chart showing MSLQ results for Student 4.

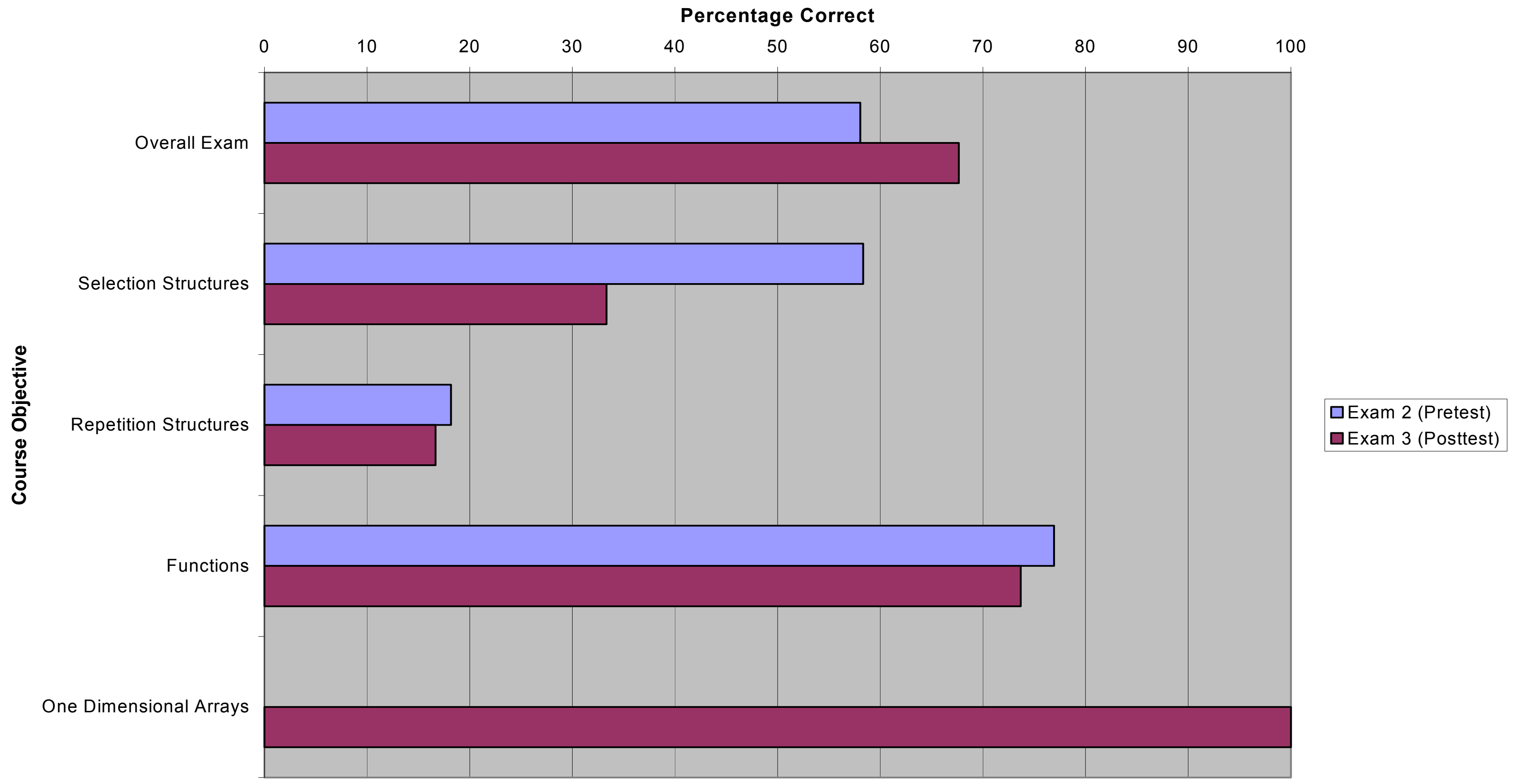


Figure 13: Bar chart showing mastery of course objectives for Student 4.

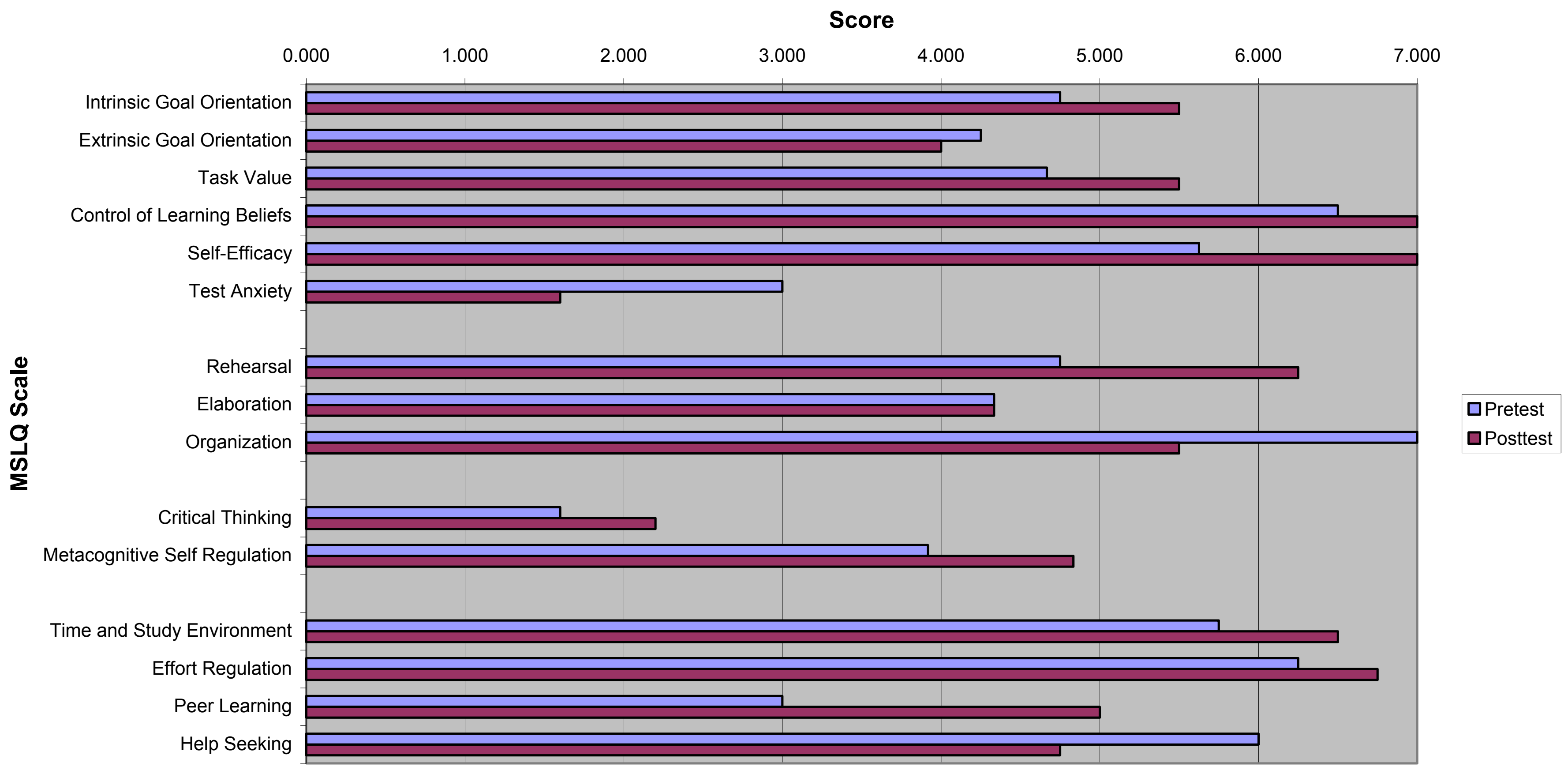


Figure 14: Bar chart showing MSLQ results for Student 5.

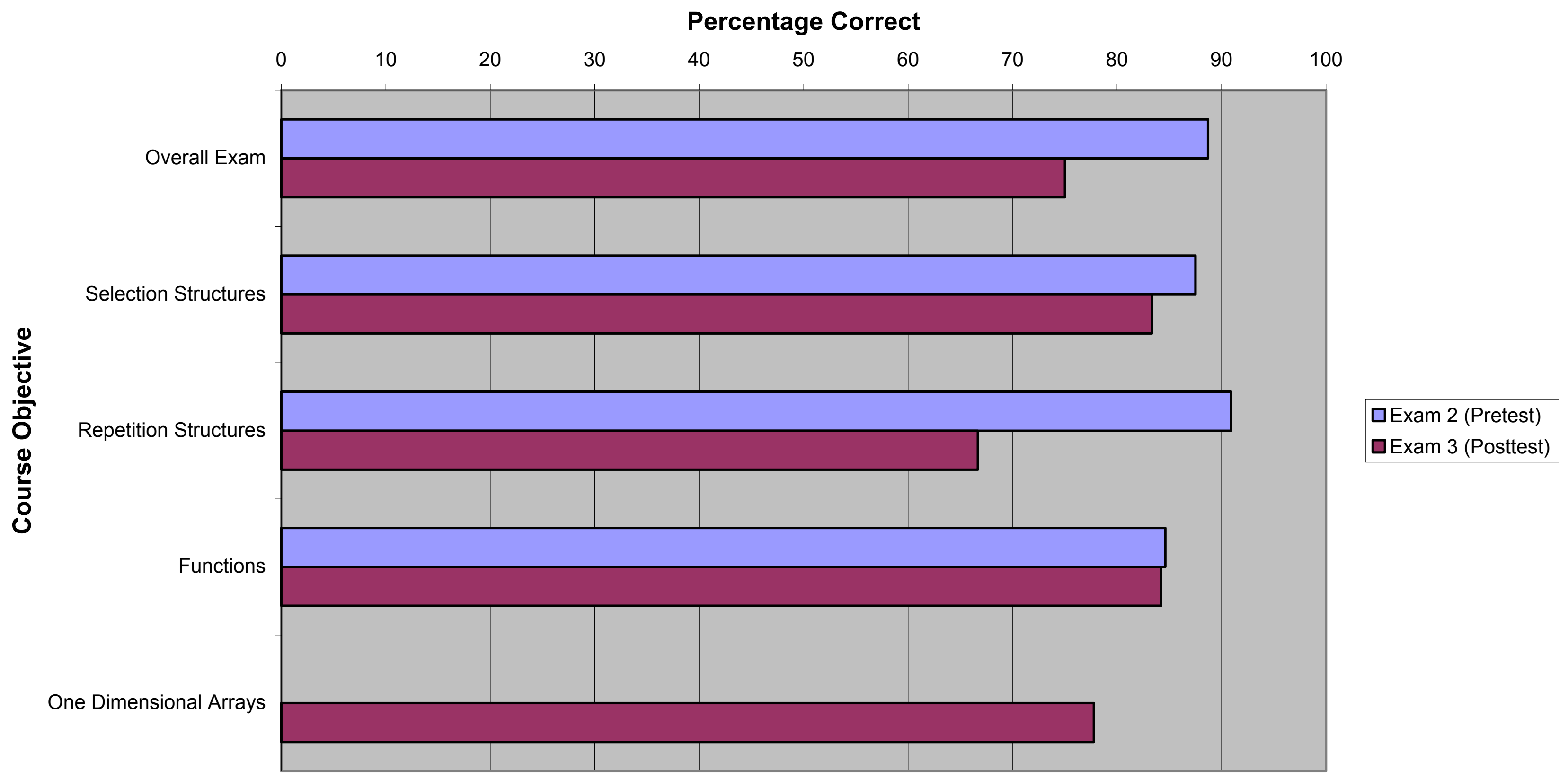
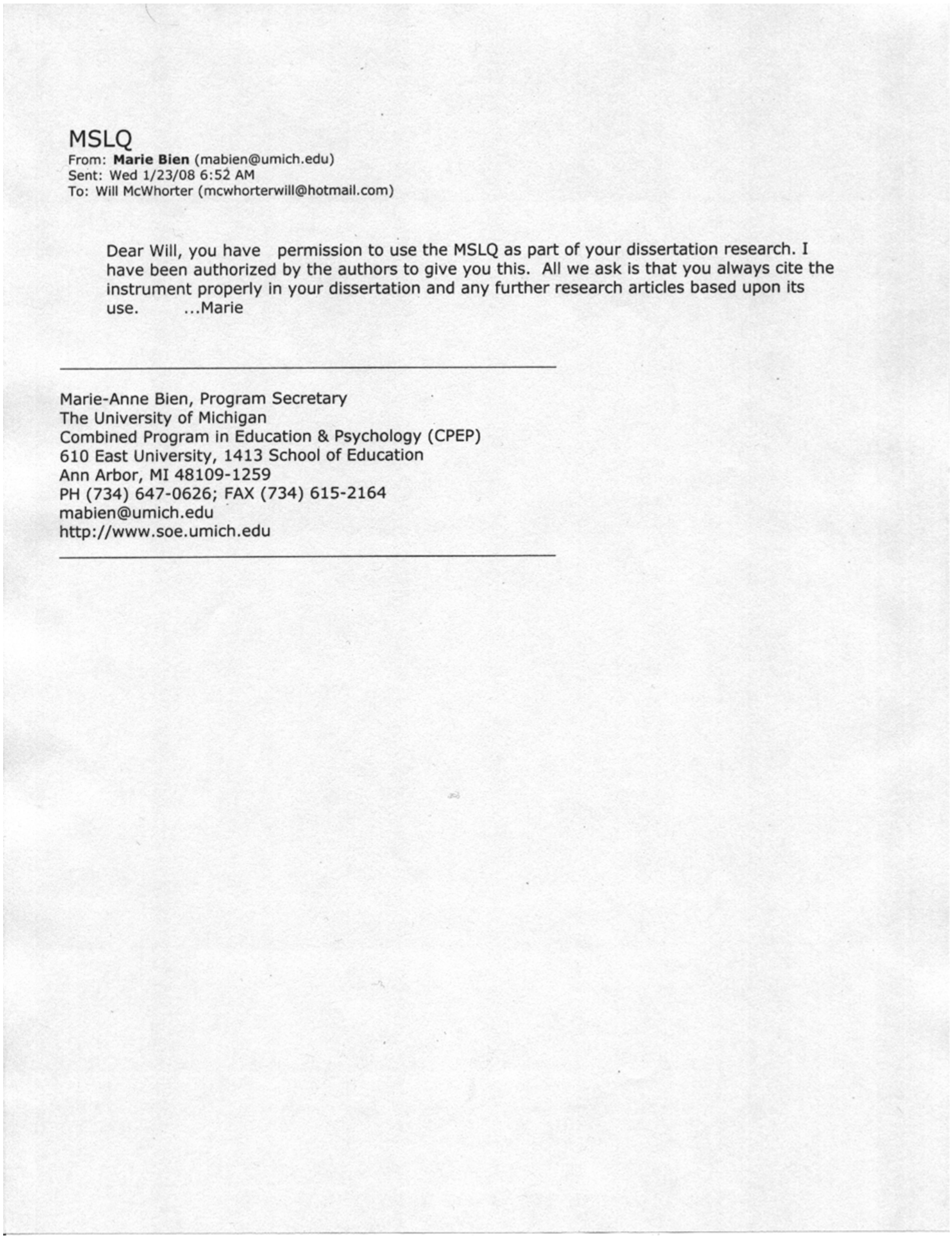


Figure 15: Bar chart showing mastery of course objectives for Student 5.

APPENDIX B

APPROVAL TO USE THE MOTIVATED STRATEGIES FOR LEARNING QUESTIONNAIRE



MSLQ
From: **Marie Bien** (mabien@umich.edu)
Sent: Wed 1/23/08 6:52 AM
To: Will McWhorter (mcwhorterwill@hotmail.com)

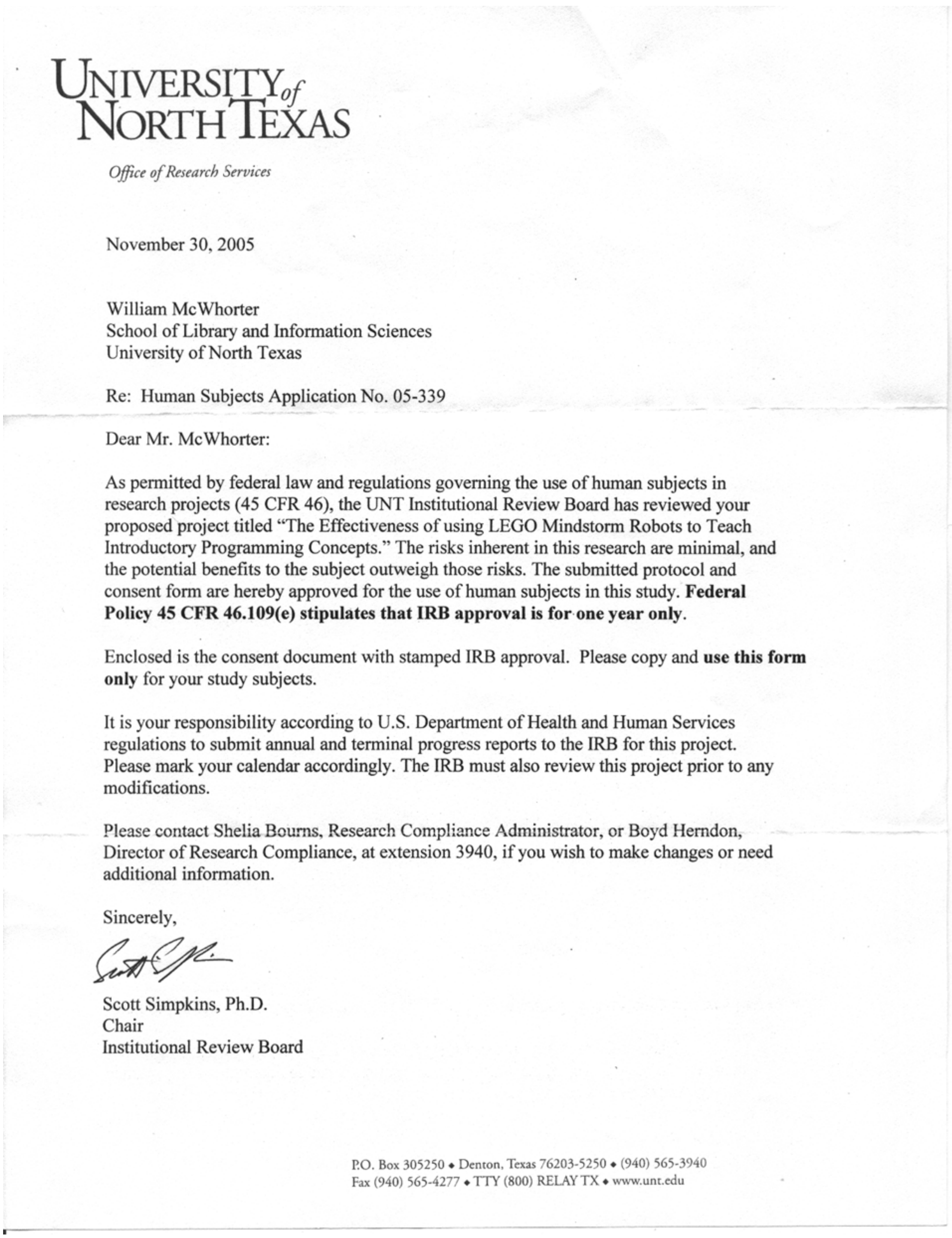
Dear Will, you have permission to use the MSLQ as part of your dissertation research. I have been authorized by the authors to give you this. All we ask is that you always cite the instrument properly in your dissertation and any further research articles based upon its use. ...Marie

---

Marie-Anne Bien, Program Secretary
The University of Michigan
Combined Program in Education & Psychology (CPEP)
610 East University, 1413 School of Education
Ann Arbor, MI 48109-1259
PH (734) 647-0626; FAX (734) 615-2164
mabien@umich.edu
http://www.soe.umich.edu

APPENDIX C

IRB APPROVAL AND INFORMED CONCENT FORMS


UNIVERSITY of NORTH TEXAS

Office of Research Services

November 30, 2005

William McWhorter
School of Library and Information Sciences
University of North Texas

Re: Human Subjects Application No. 05-339

Dear Mr. McWhorter:

As permitted by federal law and regulations governing the use of human subjects in research projects (45 CFR 46), the UNT Institutional Review Board has reviewed your proposed project titled "The Effectiveness of using LEGO Mindstorm Robots to Teach Introductory Programming Concepts." The risks inherent in this research are minimal, and the potential benefits to the subject outweigh those risks. The submitted protocol and consent form are hereby approved for the use of human subjects in this study. **Federal Policy 45 CFR 46.109(e) stipulates that IRB approval is for one year only**.

Enclosed is the consent document with stamped IRB approval. Please copy and **use this form only** for your study subjects.

It is your responsibility according to U.S. Department of Health and Human Services regulations to submit annual and terminal progress reports to the IRB for this project. Please mark your calendar accordingly. The IRB must also review this project prior to any modifications.

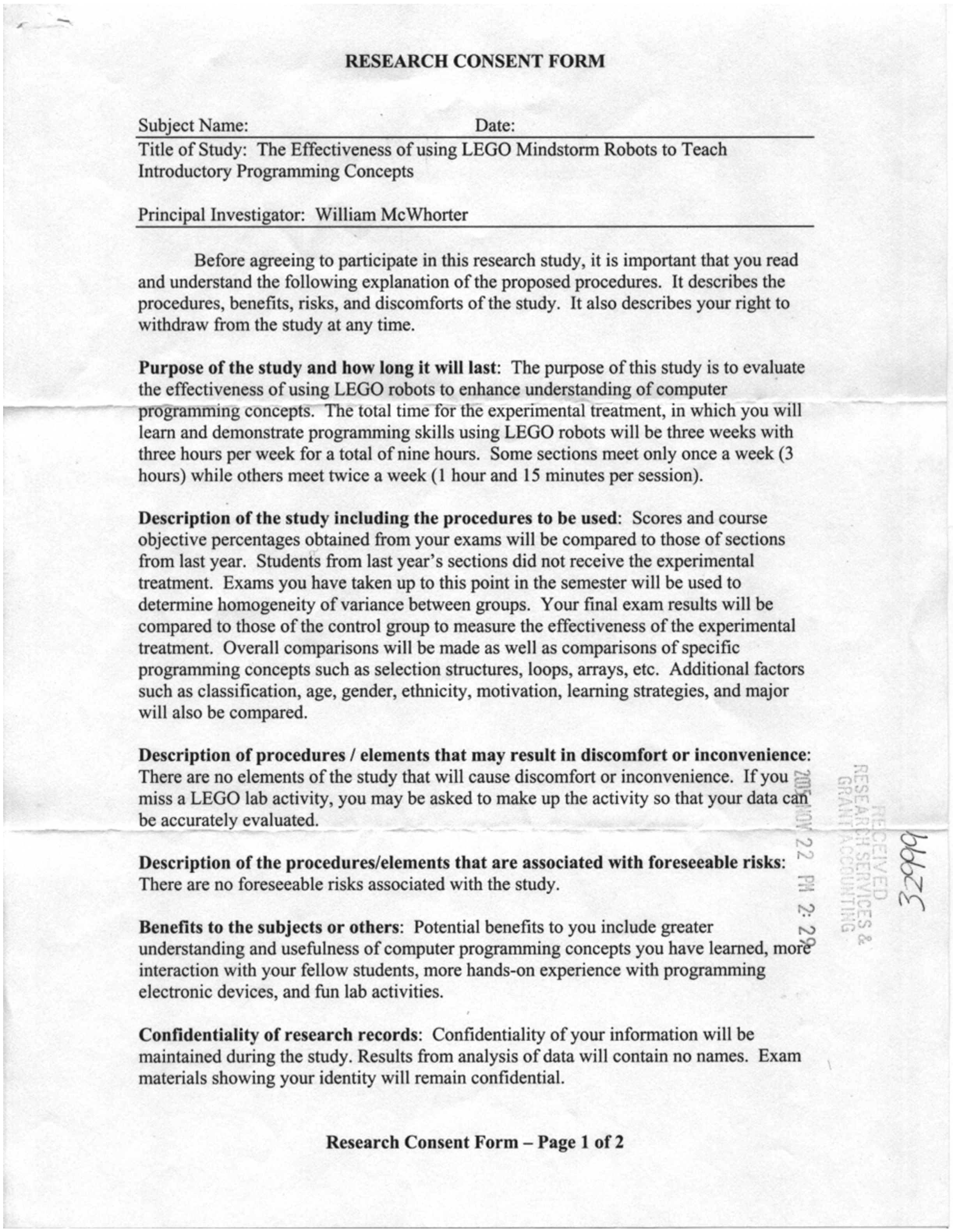
Please contact Shelia Bourns, Research Compliance Administrator, or Boyd Herndon, Director of Research Compliance, at extension 3940, if you wish to make changes or need additional information.

Sincerely,

[illegible]

Scott Simpkins, Ph.D.
Chair
Institutional Review Board

P.O. Box 305250 ♦ Denton, Texas 76203-5250 ♦ (940) 565-3940
Fax (940) 565-4277 ♦ TTY (800) RELAY TX ♦ www.unt.edu



RESEARCH CONSENT FORM

Subject Name: Date:

Title of Study: The Effectiveness of using LEGO Mindstorm Robots to Teach Introductory Programming Concepts

Principal Investigator: William McWhorter

Before agreeing to participate in this research study, it is important that you read and understand the following explanation of the proposed procedures. It describes the procedures, benefits, risks, and discomforts of the study. It also describes your right to withdraw from the study at any time.

Purpose of the study and how long it will last: The purpose of this study is to evaluate the effectiveness of using LEGO robots to enhance understanding of computer programming concepts. The total time for the experimental treatment, in which you will learn and demonstrate programming skills using LEGO robots will be three weeks with three hours per week for a total of nine hours. Some sections meet only once a week (3 hours) while others meet twice a week (1 hour and 15 minutes per session).

Description of the study including the procedures to be used: Scores and course objective percentages obtained from your exams will be compared to those of sections from last year. Students from last year's sections did not receive the experimental treatment. Exams you have taken up to this point in the semester will be used to determine homogeneity of variance between groups. Your final exam results will be compared to those of the control group to measure the effectiveness of the experimental treatment. Overall comparisons will be made as well as comparisons of specific programming concepts such as selection structures, loops, arrays, etc. Additional factors such as classification, age, gender, ethnicity, motivation, learning strategies, and major will also be compared.

Description of procedures / elements that may result in discomfort or inconvenience: There are no elements of the study that will cause discomfort or inconvenience. If you miss a LEGO lab activity, you may be asked to make up the activity so that your data can be accurately evaluated.

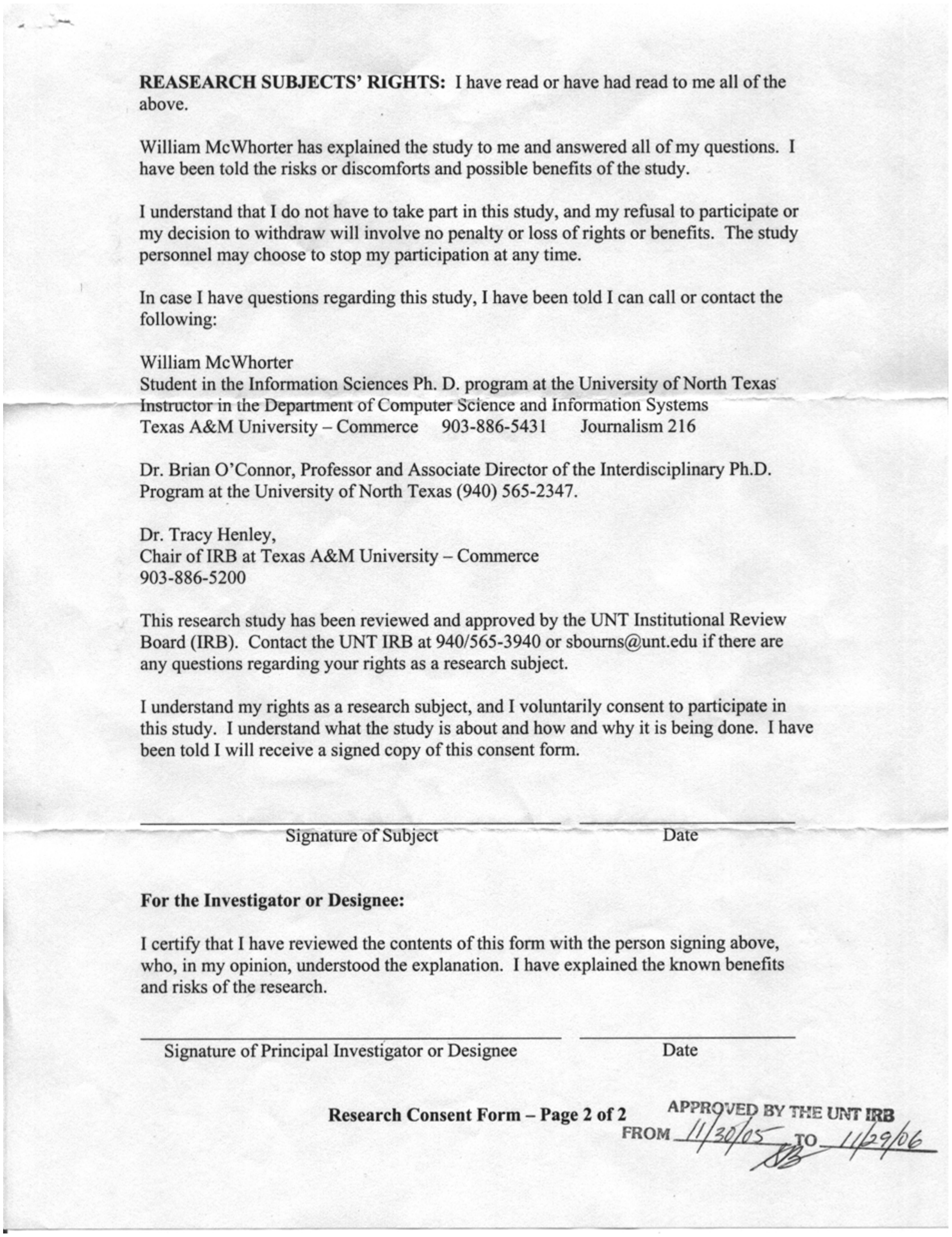
Description of the procedures/elements that are associated with foreseeable risks: There are no foreseeable risks associated with the study.

Benefits to the subjects or others: Potential benefits to you include greater understanding and usefulness of computer programming concepts you have learned, more interaction with your fellow students, more hands-on experience with programming electronic devices, and fun lab activities.

Confidentiality of research records: Confidentiality of your information will be maintained during the study. Results from analysis of data will contain no names. Exam materials showing your identity will remain confidential.

32996

Research Consent Form – Page 1 of 2



REASEARCH SUBJECTS' RIGHTS: I have read or have had read to me all of the above.

William McWhorter has explained the study to me and answered all of my questions. I have been told the risks or discomforts and possible benefits of the study.

I understand that I do not have to take part in this study, and my refusal to participate or my decision to withdraw will involve no penalty or loss of rights or benefits. The study personnel may choose to stop my participation at any time.

In case I have questions regarding this study, I have been told I can call or contact the following:

William McWhorter
Student in the Information Sciences Ph. D. program at the University of North Texas
Instructor in the Department of Computer Science and Information Systems
Texas A&M University – Commerce 903-886-5431 Journalism 216

Dr. Brian O'Connor, Professor and Associate Director of the Interdisciplinary Ph.D. Program at the University of North Texas (940) 565-2347.

Dr. Tracy Henley,
Chair of IRB at Texas A&M University – Commerce
903-886-5200

This research study has been reviewed and approved by the UNT Institutional Review Board (IRB). Contact the UNT IRB at 940/565-3940 or sbourns@unt.edu if there are any questions regarding your rights as a research subject.

I understand my rights as a research subject, and I voluntarily consent to participate in this study. I understand what the study is about and how and why it is being done. I have been told I will receive a signed copy of this consent form.

Signature of Subject	Date

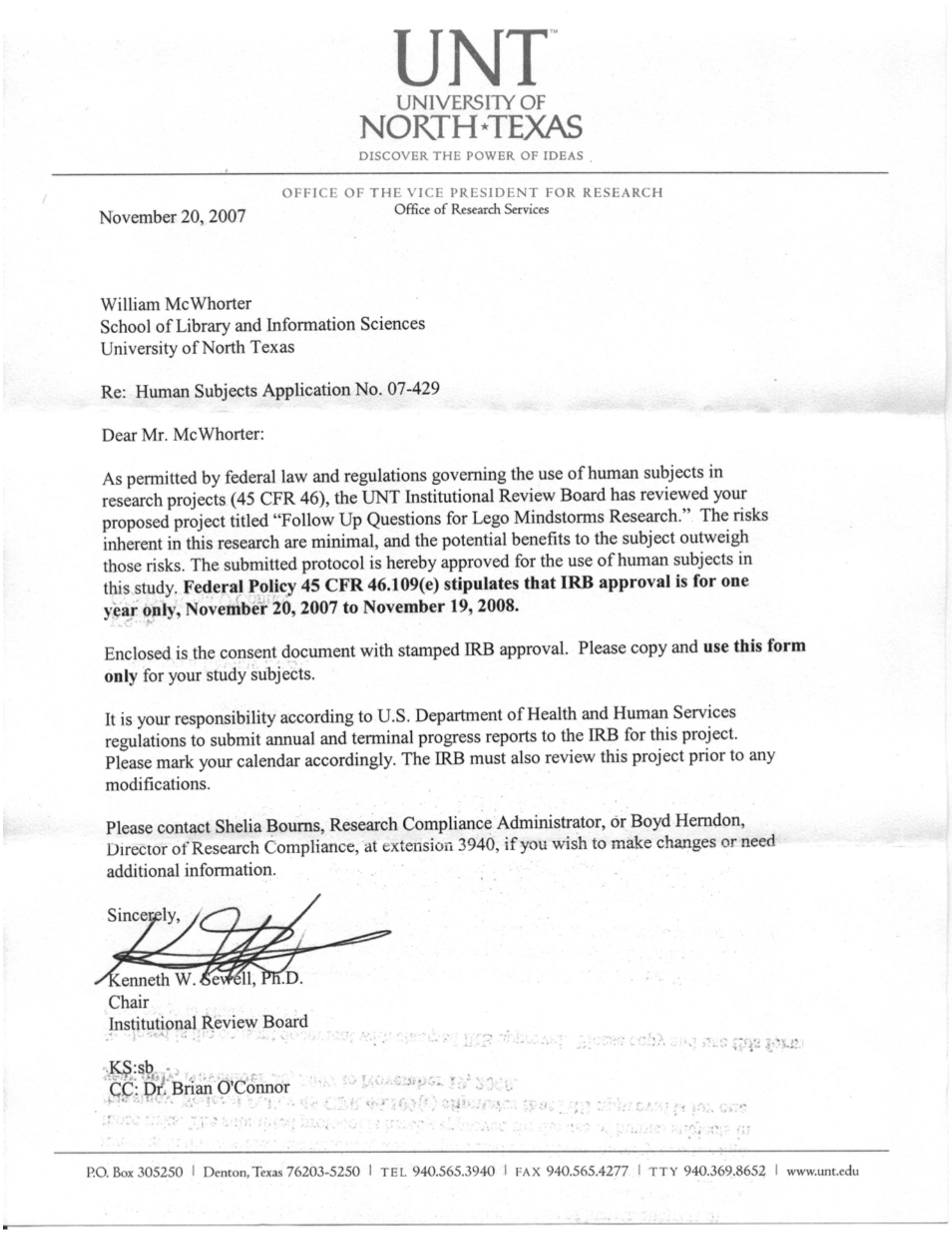
For the Investigator or Designee:

I certify that I have reviewed the contents of this form with the person signing above, who, in my opinion, understood the explanation. I have explained the known benefits and risks of the research.

Signature of Principal Investigator or Designee	Date

Research Consent Form – Page 2 of 2

APPROVED BY THE UNT IRB
FROM 11/30/05 TO 11/29/06
SB



UNT™
UNIVERSITY OF
NORTH★TEXAS
DISCOVER THE POWER OF IDEAS

OFFICE OF THE VICE PRESIDENT FOR RESEARCH
Office of Research Services

November 20, 2007

William McWhorter
School of Library and Information Sciences
University of North Texas

Re: Human Subjects Application No. 07-429

Dear Mr. McWhorter:

As permitted by federal law and regulations governing the use of human subjects in research projects (45 CFR 46), the UNT Institutional Review Board has reviewed your proposed project titled "Follow Up Questions for Lego Mindstorms Research." The risks inherent in this research are minimal, and the potential benefits to the subject outweigh those risks. The submitted protocol is hereby approved for the use of human subjects in this study. **Federal Policy 45 CFR 46.109(e) stipulates that IRB approval is for one year only, November 20, 2007 to November 19, 2008.**

Enclosed is the consent document with stamped IRB approval. Please copy and **use this form only** for your study subjects.

It is your responsibility according to U.S. Department of Health and Human Services regulations to submit annual and terminal progress reports to the IRB for this project. Please mark your calendar accordingly. The IRB must also review this project prior to any modifications.

Please contact Shelia Bourns, Research Compliance Administrator, or Boyd Herndon, Director of Research Compliance, at extension 3940, if you wish to make changes or need additional information.

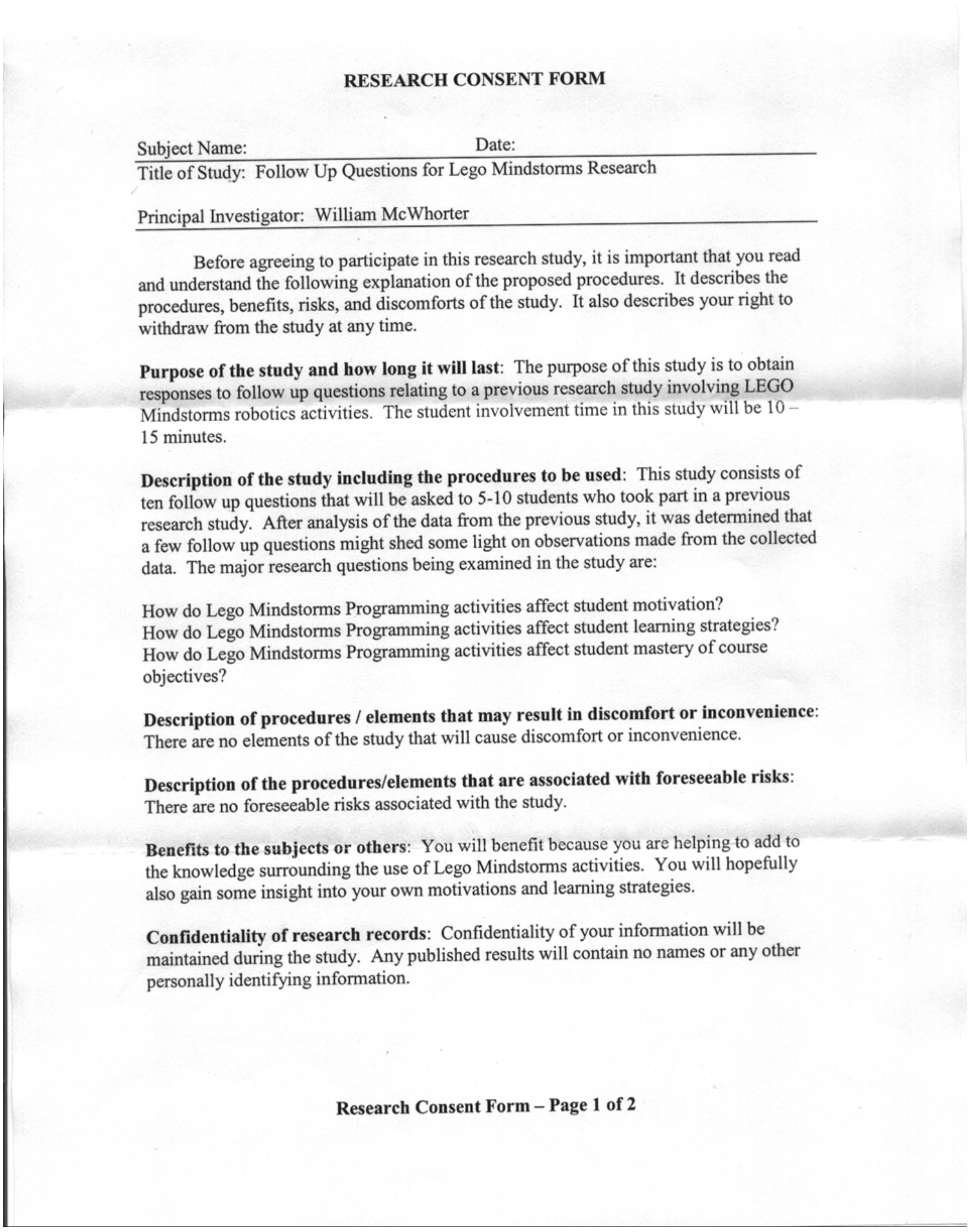
Sincerely,

[illegible]

Kenneth W. Sewell, Ph.D.
Chair
Institutional Review Board

KS:sb
CC: Dr. Brian O'Connor

P.O. Box 305250 | Denton, Texas 76203-5250 | TEL 940.565.3940 | FAX 940.565.4277 | TTY 940.369.8652 | www.unt.edu



RESEARCH CONSENT FORM

Subject Name: Date:

Title of Study: Follow Up Questions for Lego Mindstorms Research

Principal Investigator: William McWhorter

Before agreeing to participate in this research study, it is important that you read and understand the following explanation of the proposed procedures. It describes the procedures, benefits, risks, and discomforts of the study. It also describes your right to withdraw from the study at any time.

Purpose of the study and how long it will last: The purpose of this study is to obtain responses to follow up questions relating to a previous research study involving LEGO Mindstorms robotics activities. The student involvement time in this study will be 10 – 15 minutes.

Description of the study including the procedures to be used: This study consists of ten follow up questions that will be asked to 5-10 students who took part in a previous research study. After analysis of the data from the previous study, it was determined that a few follow up questions might shed some light on observations made from the collected data. The major research questions being examined in the study are:

How do Lego Mindstorms Programming activities affect student motivation?
How do Lego Mindstorms Programming activities affect student learning strategies?
How do Lego Mindstorms Programming activities affect student mastery of course objectives?

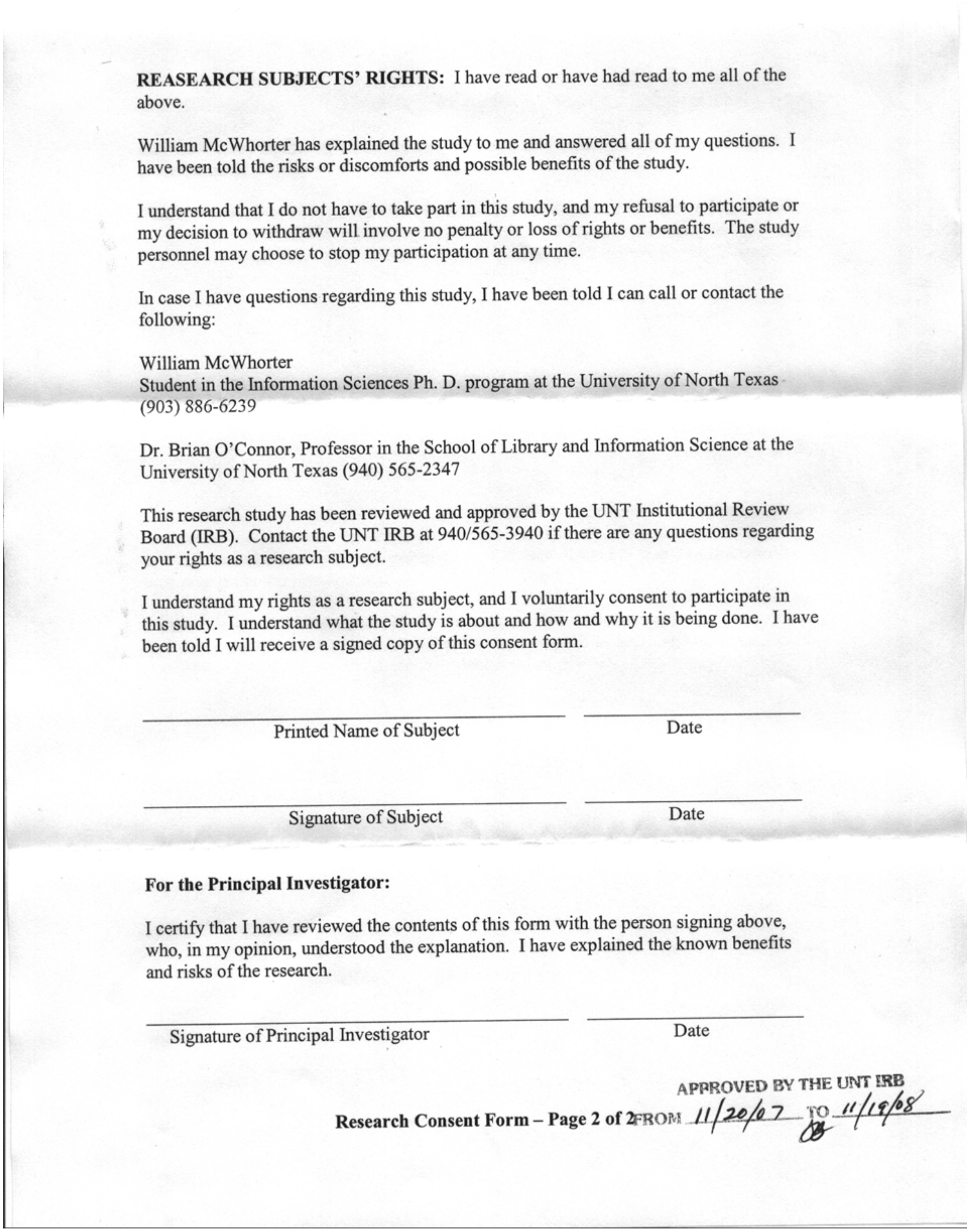
Description of procedures / elements that may result in discomfort or inconvenience: There are no elements of the study that will cause discomfort or inconvenience.

Description of the procedures/elements that are associated with foreseeable risks: There are no foreseeable risks associated with the study.

Benefits to the subjects or others: You will benefit because you are helping to add to the knowledge surrounding the use of Lego Mindstorms activities. You will hopefully also gain some insight into your own motivations and learning strategies.

Confidentiality of research records: Confidentiality of your information will be maintained during the study. Any published results will contain no names or any other personally identifying information.

Research Consent Form – Page 1 of 2



REASEARCH SUBJECTS' RIGHTS: I have read or have had read to me all of the above.

William McWhorter has explained the study to me and answered all of my questions. I have been told the risks or discomforts and possible benefits of the study.

I understand that I do not have to take part in this study, and my refusal to participate or my decision to withdraw will involve no penalty or loss of rights or benefits. The study personnel may choose to stop my participation at any time.

In case I have questions regarding this study, I have been told I can call or contact the following:

William McWhorter
Student in the Information Sciences Ph. D. program at the University of North Texas
(903) 886-6239

Dr. Brian O'Connor, Professor in the School of Library and Information Science at the University of North Texas (940) 565-2347

This research study has been reviewed and approved by the UNT Institutional Review Board (IRB). Contact the UNT IRB at 940/565-3940 if there are any questions regarding your rights as a research subject.

I understand my rights as a research subject, and I voluntarily consent to participate in this study. I understand what the study is about and how and why it is being done. I have been told I will receive a signed copy of this consent form.

Printed Name of Subject	Date
Signature of Subject	Date

For the Principal Investigator:

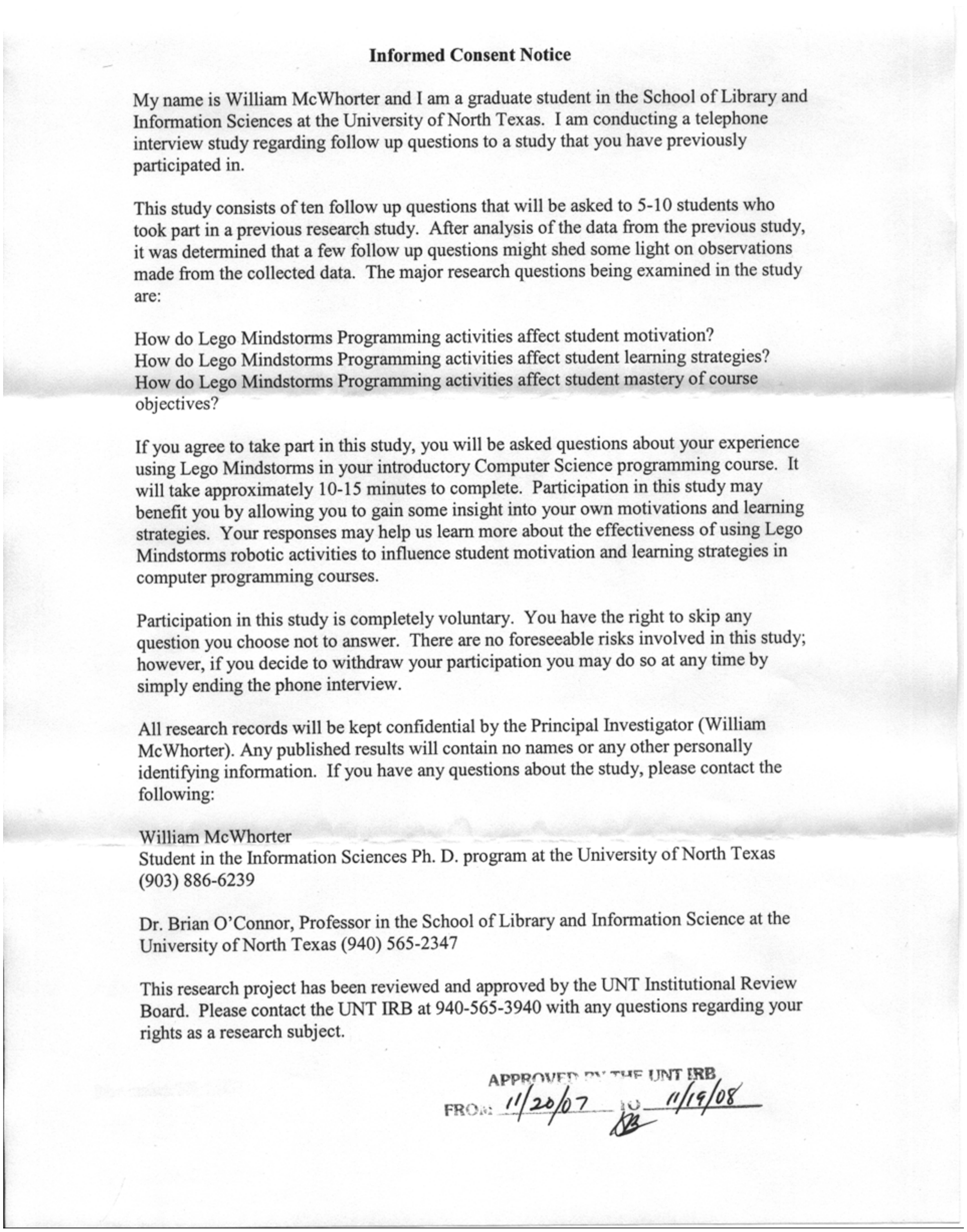
I certify that I have reviewed the contents of this form with the person signing above, who, in my opinion, understood the explanation. I have explained the known benefits and risks of the research.

Signature of Principal Investigator	Date

APPROVED BY THE UNT IRB

Research Consent Form – Page 2 of 2 FROM 11/20/07 TO 11/19/08

[illegible]



Informed Consent Notice

My name is William McWhorter and I am a graduate student in the School of Library and Information Sciences at the University of North Texas. I am conducting a telephone interview study regarding follow up questions to a study that you have previously participated in.

This study consists of ten follow up questions that will be asked to 5-10 students who took part in a previous research study. After analysis of the data from the previous study, it was determined that a few follow up questions might shed some light on observations made from the collected data. The major research questions being examined in the study are:

How do Lego Mindstorms Programming activities affect student motivation?
How do Lego Mindstorms Programming activities affect student learning strategies?
How do Lego Mindstorms Programming activities affect student mastery of course objectives?

If you agree to take part in this study, you will be asked questions about your experience using Lego Mindstorms in your introductory Computer Science programming course. It will take approximately 10-15 minutes to complete. Participation in this study may benefit you by allowing you to gain some insight into your own motivations and learning strategies. Your responses may help us learn more about the effectiveness of using Lego Mindstorms robotic activities to influence student motivation and learning strategies in computer programming courses.

Participation in this study is completely voluntary. You have the right to skip any question you choose not to answer. There are no foreseeable risks involved in this study; however, if you decide to withdraw your participation you may do so at any time by simply ending the phone interview.

All research records will be kept confidential by the Principal Investigator (William McWhorter). Any published results will contain no names or any other personally identifying information. If you have any questions about the study, please contact the following:

William McWhorter
Student in the Information Sciences Ph. D. program at the University of North Texas
(903) 886-6239

Dr. Brian O'Connor, Professor in the School of Library and Information Science at the University of North Texas (940) 565-2347

This research project has been reviewed and approved by the UNT Institutional Review Board. Please contact the UNT IRB at 940-565-3940 with any questions regarding your rights as a research subject.

APPROVED BY THE UNT IRB
FROM: 11/20/07 TO 11/19/08
[illegible]

APPENDIX D

LEGO MINDSTORMS PROGRAMMING ASSIGNMENTS AND SOLUTIONS

CSCI 151 Program 3
Light Reading Roverbot
Objectives: Functions, Loops, If statements, One-dimensional arrays

To turn in:

- Print out of your NQC code.
- Print out of your results file.
- Disk containing the NQC and results file.

You are to write a program in Not Quite C for you LEGO RoverBot. The main purpose of this robot is to collect readings from the light sensor on different surfaces. In addition, you are to keep track of the number of times the left bumper and right bumper have been pressed.

- The total time for this robot should be 25 seconds.
- If the left bumper is pressed, turn right for a quarter of a second and then continue forward. (Write a function to do this.)
- If the right bumper is pressed, turn left for a quarter of a second and then continue forward. (Write a function to do this.)
- Store each light sensor reading into a one-dimensional array. A light reading should be taken every second. There will be a total of 25 readings. (Write a separate task to do this.)
- As the robot is moving, it should play either a reoccurring sound or a song. (Write a separate task to do this.)
- When the total time is 25 seconds, the robot should stop moving. (You will need to stop the light reading task and the song playing task)
- The sum of the light reading array should then be calculated. (Write a function to do this)
- Write the number of left bumps, number of right bumps, and the light reading sum to the data log. It would also be a good idea to write the light reading values that are stored in the array to the data log.

After you have run your program on a surface, upload the data log to your computer and record the values into a text file. You will use this text file for your input on the C++ portion of this assignment. At the end of each line in the text file add the name of the surface you scanned. The text file should end up looking something like this. (Your numbers will probably be different, but you can see the format.) You can use these surfaces or other ones that you can think of. Just be reasonable. The order of the numbers on each line is the number of left bumps, number of right bumps, light sum, surface name.

```
5 7 850 Hallway Floor
4 8 800 Carpet
3 4 825 Dry Erase Board
7 2 833 Blue Mat
3 9 577 Table Top
5 7 722 Cardboard
8 5 745 Poster
```

CSCI 151 Program 4
Data Processing from Light Reading Roverbot
Objectives: Functions, Loops, If statements, One-dimensional arrays

Now that you have collected data about different surfaces from your light sensing roverbot, it would be interesting to know how the surfaces compare with one another. Write a C++ program that meets the following criteria:

- Write a function that will input the number of left bumps, number of right bumps, light sum, and surface name into separate one-dimensional arrays. Your input should be from a file that is formatted like the provided sample input.
- Write a function that will calculate the average light reading from each surface. The averages should be stored in a one-dimensional double array.
- Write a function that will calculate the total number of bumps on each surface. This should be stored in a one-dimensional integer array.
- Write a function that will calculate the overall light sum, overall number of left bumps, overall number of right bumps, overall number of total bumps, and the overall light average for all the surfaces. These should be stored into separate variables.
- Write a function that will determine the index of the brightest and darkest surface based on each surface's light sum.
- Write a function that will output your results in a table format like the provided sample output.

Sample Input File

```
5 4 567 Table Top
4 3 456 Carpet
1 2 234 Blue Mat
4 5 765 Hallway Floor
6 7 456 Sidewalk
3 2 111 Cardboard
5 5 343 Poster
```

Sample Output File

```
 Surface          LBumps RBumps TBumps LSum LAverage
----------------------------------------------------
 Table Top             5      4      9   567  28.350
 Carpet                4      3      7   456  22.800
 Blue Mat              1      2      3   234  11.700
 Hallway Floor         4      5      9   765  38.250
 Sidewalk              6      7     13   456  22.800
 Cardboard             3      2      5   111   5.550
 Poster                5      5     10   343  17.150
---------------------------------------------------
 Overall              28     28     56  2932  20.943


 Overall Brightest Surface:  Hallway Floor
 Light Sum 765
 Light Average 38.250

 Overall Darkest Surface:  Cardboard
 Light Sum 111
 Light Average 5.550
```

```
//***************************************************************************
// Will McWhorter
// will_mcwhorter@tamu-commerce.edu
// CSCI 151
// Program 3: Light Reading Rover Bot (SOLUTION PROGRAM)
// Compiler: Brick Command Center
// Language: Not Quite C
//
// The main purpose of this program is to control a LEGO robot that will
// collect readings from the light sensor on different surfaces.  It also
// keeps track of the number of times the left and right bumpers are
// pressed.   The light readings are stored in a one-dimensional array.
// Multiple functions and tasks are used in this program.  The robot also
// plays sound effects while it is moving.
//***************************************************************************

// set up names for sensors
#define LBUMP SENSOR_1
#define RBUMP SENSOR_3
#define LIGHT SENSOR_2

// set up names for motors
#define LEFT OUT_A
#define RIGHT OUT_C

// constant for the total time limit
#define TOTAL_TIME 250

#define __NOTETIME   10
#define __WAITTIME   12

// declare light_reading array
int light_reading[TOTAL_TIME/10];


//*************************************************************
// main task
// This taks sets up the sensors and starts the robot moving
// forward.  The task starts and stops two other tasks and
// calls the functions that check the left and right bumpers.
//*************************************************************
task main()
{
   // declare variables
   int left_bumps=0;
   int right_bumps=0;
   int light_sum=0;
   int time=0;

  // set up sensors
  SetSensor(LBUMP,SENSOR_TOUCH);
  SetSensor(RBUMP,SENSOR_TOUCH);
  SetSensor(LIGHT,SENSOR_LIGHT);

  // turn on motors in forward motion
  On(LEFT+RIGHT);
  // start the get_light_reading task
  start get_light_reading;

  //start the play_sound task;
  start play_sound;

  // clear the timer
  ClearTimer(0);


  // while the timer is less than or equal to the total time limit
  while(Timer(0)<=TOTAL_TIME)
  {
      // you can display the value of the time to the lcd screen like this
      time = Timer(0);
      SetUserDisplay (time, 3);

      // call the check_left function
      check_left(left_bumps);

      // call the check_right function
      check_right(right_bumps);
  }

  // turn the motors off
  Off(LEFT+RIGHT);

  // stop the get_light_reading task
  stop get_light_reading;

  // stop the play_sound task
  stop play_sound;

  // call the calc_light_sum function
  calc_light_sum(light_sum);

  // call the output_datalog function
  output_datalog(left_bumps,right_bumps,light_sum);

}

//******************************************************************
// This function checks to see if the left bumper has been pressed
// If the bumper is pressed, the robot turns right for a quarter
// of a second, increments the number of left bumps, then
// continues forward.
//******************************************************************
void check_left(int& left_bumps)
{
   if(LBUMP==1)
   {
       Rev(RIGHT);
       Wait(25);
       left_bumps++;
       Fwd(RIGHT);
   }
}
```

```
//*****************************************************************
// This function checks to see if the right bumper has been pressed
// If the bumper is pressed, the robot turns left for a quarter
// of a second, increments the number of right bumps, then
// continues forward.
//*****************************************************************
void check_right(int& right_bumps)
{
   if(RBUMP==1)
   {
       Rev(LEFT);
       Wait(25);
       right_bumps++;
       Fwd(LEFT);
   }
}

//******************************************************************
// This function adds up all the values of the light_reading
// array and stores the result into light_sum
//******************************************************************
void calc_light_sum(int& light_sum)
{
   int index=0;
   for(index=0;index<TOTAL_TIME/10;index++)
       light_sum = light_sum + light_reading[index];
}


//******************************************************************
// This function writes the results to the datalog
//******************************************************************
void output_datalog(int left_bumps, int right_bumps, int light_sum)
{
   CreateDatalog(0);
   CreateDatalog(28);
   AddToDatalog(left_bumps);
   AddToDatalog(right_bumps);
   AddToDatalog(light_sum);

   int index;
   for(index=0;index<TOTAL_TIME/10;index++)
      AddToDatalog(light_reading[index]);

}


//***********************************************************
// This task plays sound effects while the robot is moving
//***********************************************************
task play_sound()
{
  while(true)
  {
       // insert song instructions here
  }
}
```

```
//*************************************************************
// This task takes a value from the light sensor once a second
// and stores it into the light_reading array.
//*************************************************************
task get_light_reading()
{

   int index=0;
   for(index=0;index<TOTAL_TIME/10;index++)
   {
       light_reading[index]=LIGHT;
       Wait(100);
   }

}
```

```
//****************************************************************************
//  CSCI 151
//  Program 4: Rover Bot Data Processing (SOLUTION)
//  Compiler: Dev C++
//  Language: C++
//
//  This program takes the data from the light sensing rover bot and
//  calculates the light average of each surface tested.  Various totals
//  related to the bumper data are also calculated.  The brightest and
//  darkest surfaces are determined.  Input is from a file and output is
//  to a file.
//****************************************************************************

#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>

using namespace std;

// declare names constants
const int NUM_SURFACES = 7;
const int NUM_READINGS = 25;

// prototypes go here
void get_input(int[], int[], int[], string[], ifstream&);
void calc_light_average(int[], double[]);
void calc_total_bumps(int[], int[], int[]);
void calc_overall_stuff(int[], int[], int[],int&, int&, int&,int&, double&);
void determine_bright_dark(int&, int&, int[]);
void output_results(int[], int[], int[],int[], string[], double[],double,
int, int,int, int, int,int, ofstream&);


int main()
{
  // declarations
  int left_bumps[NUM_SURFACES];
  int right_bumps[NUM_SURFACES];
  int total_bumps[NUM_SURFACES];
  int light_sum[NUM_SURFACES];
  string surface_name[NUM_SURFACES];
  double light_average[NUM_SURFACES];
  double overall_light_average;
  int overall_light_sum=0;
  int overall_right_bumps=0;
  int overall_left_bumps=0;
  int overall_total_bumps=0;
  int brightest_index;
  int darkest_index;

  // declare files
  ifstream infile;
  ofstream outfile;
// open files
  infile.open("robot_input3.txt");
  outfile.open("robot_output3.txt");

  // call get_input
  get_input(left_bumps, right_bumps, light_sum, surface_name, infile);

  // call calc_light_average
  calc_light_average(light_sum,light_average);

  // call calc_total_bumps
  calc_total_bumps(left_bumps,right_bumps,total_bumps);

  // call calc_overall_stuff
calc_overall_stuff(left_bumps,right_bumps,light_sum,overall_light_sum,
         overall_left_bumps, overall_right_bumps, overall_total_bumps,
         overall_light_average);

  // call determine_bright_dark
  determine_bright_dark(brightest_index,darkest_index,light_sum);

  // call output_results
output_results(left_bumps, right_bumps, total_bumps, light_sum,
            surface_name,light_average, overall_light_average,
            overall_light_sum, overall_right_bumps, overall_left_bumps,
            overall_total_bumps, brightest_index,
            darkest_index, outfile);

  return 0;
}


//*****************************************************************************
//  This function reads the data from the input file and stores the values
//  into one-dimensional arrays.
//*****************************************************************************
void get_input(int left_bumps[], int right_bumps[], int light_sum[],
               string surface_name[], ifstream& infile)
{
   int index;
   for(index=0;index<NUM_SURFACES;index++)
   {
       infile >> left_bumps[index] >> right_bumps[index] >> light_sum[index];
       getline(infile,surface_name[index]);
   }
}


//*****************************************************************************
//  This function calculates the average light reading from each surface
//*****************************************************************************
void calc_light_average(int light_sum[], double light_average[])
{
   int index;
   for(index=0;index<NUM_SURFACES;index++)
      light_average[index]=double(light_sum[index])/double(NUM_READINGS);
}
```

```
//**************************************************************************
//  This function calculates the total bumps from each surface tested.
//**************************************************************************
void calc_total_bumps(int left_bumps[], int right_bumps[], int total_bumps[])
{
   int index;
   for(index=0;index<NUM_SURFACES;index++)
      total_bumps[index]=left_bumps[index]+right_bumps[index];
}


//***************************************************************************
// This function calculates the overall light sum, overall number of left
// bumps overall number of right bumps, overall number of bumps, and the
// overall light average.
//***************************************************************************
void calc_overall_stuff(int left_bumps[], int right_bumps[], int light_sum[],
                int& overall_light_sum, int& overall_left_bumps,
                int& overall_right_bumps, int& overall_total_bumps,
                double& overall_light_average)
{
   int index;
   for(index=0;index<NUM_SURFACES;index++)
   {
      overall_light_sum += light_sum[index];
      // overall_light_sum = overall_sum + light_sum[index];
      overall_left_bumps += left_bumps[index];
      overall_right_bumps += right_bumps[index];
   }
   overall_total_bumps = overall_left_bumps + overall_right_bumps;
   overall_light_average = double(overall_light_sum) /
                           double(NUM_READINGS*NUM_SURFACES);
}


//***************************************************************************
// This function determines the index of the brightest and darkest surfaces
// based on each surface's light sum.
//***************************************************************************
void determine_bright_dark(int& brightest_index, int& darkest_index,
                           int light_sum[])
{
   int index;
   brightest_index=0;
   darkest_index=0;
   for(index=1;index<NUM_SURFACES;index++)
   {
      if(light_sum[index] > light_sum[brightest_index])
          brightest_index = index;

      if(light_sum[index] < light_sum[darkest_index])
          darkest_index = index;

   }
}
```

```
//***************************************************************************
//  This function outputs a summary of results in a table format.  The output
//  is to a file.
//***************************************************************************
void output_results(int left_bumps[], int right_bumps[], int total_bumps[],
       int light_sum[], string surface_name[], double light_average[],
       double overall_light_average, int overall_light_sum,
       int overall_right_bumps, int overall_left_bumps,
       int overall_total_bumps, int brightest_index, int darkest_index,
       ofstream& outfile)
{
    int index;

    // set up floating up format
    outfile << fixed << showpoint << setprecision(3);

    // output the headings
    outfile << left << setw(17) << " Surface" << setw(7) << "LBumps"
            << setw(7) << "RBumps" << setw(7) << "TBumps"
            << setw(5) << "LSum" << setw(9) << "LAverage" << endl;
    outfile << "-----------------------------------------------------"
            << endl;

    // this loops outputs one row at a time
    for(index=0;index<NUM_SURFACES;index++)
    {
        outfile << left << setw(14) << surface_name[index]
                << right << setw(7) << left_bumps[index]
                << setw(7) << right_bumps[index]
                << setw(7) << total_bumps[index]
                << setw(7) << light_sum[index]
                << setw(8) << light_average[index]
                << endl;
    }

    outfile << "-----------------------------------------------------" << endl;

    // output overall stats
    outfile << left << setw(14) << " Overall"
            << right << setw(7) << overall_left_bumps
            << setw(7) << overall_right_bumps
            << setw(7) << overall_total_bumps
            << setw(7) << overall_light_sum
            << setw(8) << overall_light_average
            << endl;

    outfile << endl << endl;
    // output brightest and darkest surfaces information
    outfile << "Overall Brightest Surface: " << surface_name[brightest_index]
            << endl;
    outfile << "Light Sum " << light_sum[brightest_index] << endl;
    outfile << "Light Average " << light_average[brightest_index]
            << endl << endl;

    outfile << "Overall Darkest Surface: " << surface_name[darkest_index]
            << endl;
    outfile << "Light Sum " << light_sum[darkest_index] << endl;
    outfile << "Light Average " << light_average[darkest_index]
            << endl << endl;

}
```

REFERENCES

Anderson, E. F., & McLoughlin, L. (2007). Critters in the classroom: A 3D computer-game-like tool for teaching programming to computer animation students. *International Conference on Computer Graphics and Interactive Techniques: ACM SIGGRAPH 2007 Educators Program*, article no. 7.

Anderson, R. O., Welch, W., & Harris, L. (1983). Computer inequitities in opportunities for computer literacy. *Department of Sociology, University of Minnesota*.

Bandura, A. (1986). *Social foundations of thought and action: A social cognitive theory*. Englewood Cliffs, NJ: Prentice-Hall.

Barnes, D. J. (2002). Teaching introductory Java through LEGO MINDSTORMS models. *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education*, 147-151.

Baum, D. (2000). *Definitive guide to Lego Mindstorms*. Apress.

Bergin, S., & Reilly, R. (2005). The influence of motivation and comfort-level on learning to program. *Proceedings of the 17th Workshop of the Psychology of Programming Interest Group*, 293-304.

Bergin, S., Reilly, R., & Traynor, D. (2005). Examining the role of self-regulated learning on introductory programming performance. *Proceedings of the 2005 International Workshop on Computing Education Research*, 81-86.

Bierre, K. J., & Phelps, A. M. (2004). The use of MUPPETS in an introductory java programming course. *Proceedings of the 5th Conference on Information Technology Education*, 122-127.

Bloom, B., Mesia, B., & Krathwohl, D. (1964). *Taxonomy of educational objectives*. New York: David McKay.

BlueJ - teaching Java - learning Java. (n.d.). Retrieved January 1, 2008, from http://www.bluej.org/

Brick Command Center. (2002). Retrieved July 1, 2004, from http://bricxcc.sourceforge.net/

Carlisle, M., Wilson, T., Humphries, J., & Moore, J. (n.d.). *RAPTOR flowchart interpreter*. Retrieved January 1, 2008, from http://raptor.martincarlisle.com/

Cliburn, D. C. (2006). A CS0 course for the liberal arts. *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, 77-81.

Cooper, S., Dann, W., & Pausch, R. (2000). Alice: A 3-D tool for introductory programming concepts. *Proceedings of the Fifth Annual CCSC Northeastern Conference on the Journal of Computing in Small Colleges*, 107-116.

Fagin, B., & Merkle, L. (2003). Measuring the effectiveness of robots in teaching computer science. *ACM SIGCSE Bulletin, 35*(1), 307-311.

Fagin, B. S., & Merkle, L. (2002). Quantitative analysis of the effects of robots on introductory Computer Science education. *Journal on Educational Resources in Computing, 2*(4), 1-18.

Fisher, A., Margolis, J., & Miller, F. (1997). Undergraduate women in computer science: Experience, motivation and culture. *ACM SIGCSE Bulletin, 29*(1), 106-110.

Gall, M. D., Gall, J. P., & Borg, W. R. (1996). *Educational research: An introduction* (6th ed.). Boston: Allyn and Bacon.

Garcia, M. A., & McNeill, H. P. (2002). Learning how to develop software using the toy LEGO Mindstorms (poster session). *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education*, 239.

Gardner, H. (1983). *Frames of mind: The theory of multiple intelligences*. New York: Basic Books.

Gross, P., & Powers, K. (2005). Evaluating assessments of novice programming environments. *Proceedings of the 2005 International Workshop on Computing Education Research*, 99-110.

Hinkle, D. E., Wiersma, W., & Jurs, S. G. (2003). *Applied statistics for the behavioral sciences* (5th ed.). Boston: Houghton Mifflin Company.

Hodgins, H.W. & Connor, M.L. (2000). Everything you ever wanted to know about learning styles but were afraid to ask. *Learning in the New Economy Magazine*, Fall 2000.

Jenkins, T. (2001). The motivation of students of programming. *ACM SIGCSE Bulletin, 33*(3), 53-56.

Jeliot. (n.d.). Retrieved January 1, 2008, from http://cs.joensuu.fi/jeliot/

Klassner, F., & Anderson, S. D. (2003). LEGO Mindstorms: Not just for K-12 anymore. *IEEE Robotics and Automation Magazine, 10*(2), 12-18.

Konstam, A., & Howland, J. E. (1994). Teaching computer science principles to liberal arts students using Scheme. *ACM SIGCSE Bulletin, 26*(4), 29-34,40.

Lawhead, P. B., Duncan, M. E., Bland, C. G., Goldweber, M., Schep, M., Barnes, D. J. et al. (2002). A road map for teaching introductory programming using LEGO© Mindstorms robots. *Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, 191-201.

LEGO Group. (1999). *Robotics Invention System 2.0 Constructopedia*.

Leutenegger, S., & Edgington, J. (2007). A games first approach to teaching introductory programming. *ACM SIGCSE Bulletin, 39*(1), 115-118.

Mahmoud, Q. H., Dobosiewicz, W., & Swayne, D. (2004). Redesigning introductory computer programming with HTML, JavaScript, and Java. *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, 120-124.

Malik, D. S. (2007). *C++ programming: From problem analysis to program design* (3rd ed.). Boston: Thomson Course Technology.

McNally, M., Goldweber, M., Fagin, B., & Klassner, F. (2006). Do LEGO Mindstorms robots have a future in CS education? *ACM SIGCSE Bulletin, 38*(1), 61-62.

McNeill, H. P., & Binkerd, C. L. (2001). Resources for using LEGO Mindstorms. *Proceedings of the Seventh Annual Consortium for Computing in Small Colleges*, 48-55.

Overmars, M. (1999). Programming LEGO robots using NQC. *Utrecht University, the Netherlands*. Retrieved July 1, 2004, from http://people.cs.uu.nl/markov/lego/

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Papert, S. (1980s). *Construcionism vs. Instructionism*. Speech presented at Japan Educators Conference, Japan (via video).

Papert, S. (1993). *The children's machine: Rethinking school in the age of the computer*. New York: Basic Books.

Papert, S. (1999). What is Logo? And who needs it? In *Logo philosophy and implementation* (pp. IX-XVI) [Introduction]. Logo Computer Systems.

Pattis, R. E. (1995). *Karel the robot: A gentle introduction to the art of programming* (2nd ed.). New York: John Wiley & Sons. (Original work published 1981)

Pea, R. D., & Sheingold, K. (1987). *Mirrors of minds: Patterns of experience in educational computing*. Norwood, NJ: Ablex.

Phelps, A. M., Bierre, K. J., & Parks, D. M. (2003). MUPPETS: Multi-user programming pedagogy for enhancing traditional study. *Proceedings of the 4th Conference on Information Technology Curriculum*, 100-105.

Pintrich, P. R. (1999). The role of motivation in promoting and sustaining self-regulated learning. *International Journal of Educational Research, 31*, 459-470.

Pintrich, P. R. (2000). Multiple goals, multiple pathways: The role of goal orientation in learning and achievement. *Journal of Educational Psychology, 92*(3), 544-555.

Pintrich, P. R. (2002). The role of metacognitive knowledge in learning, teaching, and assessing. *Theory Into Practice, 41*(4), 219-225.

Pintrich, P. R., & DeGroot, E. V. (1990). Motivational and self-regulated learning components of classroom academic performance. *Journal of Educational Psychology, 82*(1), 33-40.

Pintrich, P. R., Smith, D. A., Garcia, T., & McKeachie, W. J. (1991). *A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ)* (Technical Report No. 91-B-004). The Regents of the University of Michigan.

Powers, K., Gross, P., Cooper, S., McNally, M., Goldman, K. J., Proulx, V. et al. (2006). Tools for teaching introductory programming: What works? *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, 560-561.

Rudestam, K. E., & Newton, R. R. (1992). *Surviving your dissertation: A comprehensive guide to content and process*. Newbury Park, CA: Sage.

Tharp, A. L. (1987). Let's motivate. *ACM SIGCSE Bulletin, 19*(1), 415-422.

Turing's Craft. (n.d.). Retrieved January 1, 2008, from http://www.turingscraft.com/

Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: A study of twelve factors. *ACM SIGCSE Bulletin, 33*(1), 184-188.

Wolz, U. (2001). Teaching design and project management with Lego RCX robots. *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education*, 95-99.

Wong, K.W. (2001). Teaching programming with LEGO RCX robots. *Proceedings of ISECON, 18*.

UNCITED REFERENCES

Becker, B. W. (2001). Teaching CS1 with karel the robot in Java. *ACM SIGCSE Bulletin, 33*(1), 50-54.

Berman, A. M. (1994). Does Scheme enhance an introductory programming course?: Some preliminary empirical results. *ACM SIGPLAN Notices, 29*(2), 44-48.

Buck, D., & Stucki, D. J. (2001). JKarelRobot: A case study in supporting levels of cognitive development in the computer science curriculum. *ACM SIGCSE Bulletin, 33*(1), 16-20.

Carver, C. A., Howard, R. A., & Lane, W. D. (1996). A methodology for active, student-controlled learning: Motivating our weakest students. *Proceedings of the Twenty-seventh SIGCSE Technical Symposium on Computer Science Education*, 195-199.

Clements, D. H., & Meredith, J. S. (1993). Research on Logo: Effects and efficacy. *Journal of Computing in Childhood Education, 4*(3-4), 263-290.

Dann, W., Cooper, S., Pausch, R., & Slater, D. (n.d.). *Learning to program with Alice*. Retrieved November 14, 2007, from http://www.aliceprogramming.net/

Fagin, B. (2000). An Ada interface to Lego Mindstorms. *ACM SIGAda Ada Letters, XX*(3), 20-40.

Fagin, B. (2000). Using Ada-based robotics to teach computer science. *ACM SIGCSE Bulletin, 32*(3), 148-151.

Fagin, B. S., Merkle, L. D., & Eggers, T. W. (2001). Teaching computer science with robotics using Ada/Mindstorms 2.0. *Proceedings of the 2001 Annual ACM SIGAda International Conference on Ada*, 73-78.

Gibbs, D. C. (2000). The effect of a constructivist learning environment for field-dependent/independent students on achievement in introductory computer programming. *ACM SIGCSE Bulletin, 32*(1), 207-211.

Goldwebber, M., Congdon, C., Fagin, B., Hwang, D., & Klassner, F. (2001). The use of robots in the undergraduate curriculum: Experience reports. *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education*, 404-405.

Imberman, S. P., & Klibaner, R. (2005). A robotics lab for CS1. *Journal of Computing Sciences in Colleges, 21*(2), 131-137.

Kay, J. S. (2003). Teaching robotics from a computer science perspective. *Journal of Computing Sciences in Colleges, 19*(2), 329-336.

Kelleher, C., Pausch, R., & Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1455-1464.

Knudsen, J. B. (1999). *The unofficial guide to Lego Mindstorms robots*. Sebastopol, CA: O'Reilly & Associates.

Kushan, B. (1994). Preparing programming teachers. *Proceedings of the Twenty-fifth SIGCSE Symposium on Computer Science Education*, 248-252.

Lawhead, P., Duncan, M. E., Bland, C. G., Goldweber, M., Schep, M., & Barnes, D. J. (2003). Legos, Java and programming assignments for CS1. *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, 47-48.

LEGO Group. (n.d.). *Official LEGO MINDSTORMS site*. Retrieved April 16, 2005, from LEGO Web site: http://mindstorms.LEGO.com

Leska, C. (2004). Introducing undergraduates to programming using robots in the general education curriculum (poster session). *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 263.

Linnenbrink, E. A., & Pintrich, P. R. (2002). Motivation as an enabler for academic success. *School Psychology Review, 31*(3), 313-327.

Masui, T. (2000). Real-world programming. *Proceedings of DARE 2000 on Designing Augmented Reality Environments*, 115-120.

Maxwell, S. E., & Delaney, H. D. (2000). *Designing experiments and analyzing data: A model comparison perspective*. Mahwah, NJ: Lawrence Erlbaum Associates.

McNerney, T. S. (2004). From turtles to tangible programming bricks: Explorations in physical language design. *Personal and Ubiquitous Computing, 8*(5), 326-337.

Mitchell, M., Sheard, J., & Markham, S. (2000). Student motivation and positive impressions of computing subjects. *Proceedings of the Australasian Conference on Computing Education*, 189-194.

Murphy, L., McCauley, R., & Westbrook, S. (2006). Women catch up: Gender differences in learning programming concepts. *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, 17-21.

Papert, S., & The Massachusetts Institute of Technology Media Laboratory. (1999). *Constructionism: Research reports and essays*. Norwood, NJ: Ablex.

Rich, L., & Perry, H. (2004). A CS1 course designed to address interests of women. *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, 190-194.

Rountree, N., Rountree, J., Robins, A., & Hannah, R. (2004). Interacting factors that predict success and failure in a CS1 course. *ACM SIGCSE Bulletin, 36*(4), 101-104.

Shaffer, D. (1986). The use of Logo in an introductory computer science course. *ACM SIGCSE Bulletin, 18*(4), 28-31.

VanderStoep, S. W., & Pintrich, P. R. (2003). *Learning to learn: The skill and will of college success*. Upper Saddle River, NJ: Pearson Education.

Vento, A., Beltran, C., & Taniuchi, E. (2002). A quantitative analysis of robotic languages. *Journal of Computing Sciences in Colleges, 17*(5), 72-80.

Waraich, A. (2004). Using narrative as a motivating device to teach binary arithmetic and logic gates. *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, 97-101.

Wilson, B. C. (2002). A study of factors promoting success in computer science including gender differences. *Computer Science Education, 12*(1-2), 141-164.