Illinois Mathematics & Computer Science Articulation Guide

Prepared by IMACC-ISMAA Joint Task Force

May, 2008

Table of Contents

Page

Preface i
Introduction 1
Technology Statement 4
Standards Statement 5
College Algebra Statement 5

I. Mathematics General Education Courses 6
1. General Education Statistics 7
2. General Education Mathematics 10
3. Quantitative Literacy 12
4. Elementary Mathematical Modeling 13

II. Mathematics Pretransfer (Developmental) Courses 14
1. Arithmetic 16
2. Pre-Alegebra 17
3. Basic Algebra 19
4. Geometry 21
5. Intermediate Algebra 23
5AG. Intermediate Algebra with Geometry 25
5BI. Combined Basic and Intermediate Algebra 28

III. Mathematics Courses for Mathematics, Engineering, Computer Science, and Business Majors 30
1. College Algebra 30
2. Trigonometry 32
3. College Algebra and Trigonometry 33
4. Elementary Functions (Precalculus) 34
5. Analytic Geometry-Calculus 35
6. Differential Equations 37
7. Introduction to Linear Algebra 38
8. Mathematical Statistics 39
9. Statistics (for Business Majors) 40
10. Finite Mathematics (A and B) (for Business and Management) 42
11. Calculus for Business and Social Science 44
12. Mathematics for Elementary Teaching I, II 45
13. Discrete Mathematics 47

IV. Computer Science Courses and Recommended Courses of Study 48
1. Foundations of Informational Technology (Computer Literacy) 48
2. Computer Science I 50
3. Computer Science II 52
4. Computer Science III 54
5. Computer Programming for Science and Engineering 56
6. Discrete Structures 58
7. Event Driven Programming 59
8. Computer Organization and Architecture 61
9. Computer Science Major 63
10. Engineering Computer Science Major 64
11. Mathematics, Physical Science, or Engineering (Mechanical, Electrical) Major 65
12. Business Curriculums 66

V. Additional Course Options as Recommended by the Association of Computing Machinery (ACM) 67
1. Computer Science I – Version A 68
2. Computer Science II – Version A 71
3. Computer Science III – Version A 74
4. Net Centric Operating Systems 76

Conclusion 79
Addendum 80

Preface

This fourth edition of the Illinois Mathematics and Computer Science Articulation Guide was prepared in 1994-95 and includes the descriptions of the general education courses that were recommended by the Illinois Articulation Initiative in the General Education Core Curriculum. The General Education Core Curriculum will be acceptable for students who transfer in lieu of the receiving institution's general education program.

The Articulation Guide was updated so that the general education objectives of the Illinois Articulation Initiative and the general education course descriptions in the Articulation Guide would be consistent. These revisions were again jointly developed by the Illinois Mathematics Association of Community Colleges (IMACC) and the Illinois Section of the Mathematics Association of America (ISMAA). Since there were several significant changes included in the General Education Core Curriculum for mathematics, this fourth edition of the Articulation Guide is very timely and will be very useful to all colleges and universities in Illinois that are participating in the Illinois Articulation Initiative.

Dr. Ivan J. Lach
Deputy Director for Programs
Illinois Community College Board

Introduction

The IMACC-ISMAA Joint Task Force is a joint effort of the Illinois Mathematics Association of Community Colleges and the Illinois Section of the Mathematical Association of America.

The first Articulation Guide (then called Curriculum Guide) was prepared in 1969-73 by the Junior College Task Force of the Illinois Section of the Mathematical Association of America serving as the Mathematics Advisory Board to the Illinois Community College Board. Members of the original Task Force were:

William L. Drezdzon, Chairman
Oakton Community College

John Bradburn
Elgin Community College

Lawrence Eggan
Illinois State University

Loren Pixley
Community College of Decatur

Genevieve Snider
Belleville Area College

Arnold Wendt
Western Illinois University

Dale Williams
Illinois Central College

The second edition of the Articulation Guide was prepared in 1981-1982. The second edition Task Force members were:

Arnold Wendt, Chairman
Western Illinois University

John Bradburn
Elgin Community College

Theresa Butzen
William Rainey Harper College

Ann Dice
William Rainey Harper College

William Drezdzon
Oakton Community College

Barbara Juister
Elgin Community College

John LeDuc
Eastern Illinois University

Jess Moore
John A. Logan College

The third edition of the Articulation Guide prepared in 1989-91. The third edition Task Force initially focused on the tasks of:

1. reviewing and revising courses in the Mathematics Section of the Guide
2. creating a pretransfer section to include developmental/ remedial courses, and
3. creating a General Education section.

Those third edition Task Force members were:

Dale Ewen, Chair
Parkland College

Carole Bauer
Triton College

Philip Beckman
Black Hawk College

John Bradburn
Elgin Community College

Dick Fritz
Moraine Valley Community College

David Gustafson
Rock Valley College

Robert Hathway
Illinois State University

Carla Hillman
Black Hawk College-East

Carole LaCampagne
Northern Illinois University

Tom McCabe
William Rainey
Harper College

LaVerne McFadden
Parkland College

Phillip McGill
Illinois Central College

Roy Meyerholtz
Eastern Illinois University

Anthony Peressini
University of Illinois
at Urbana-Champaign

Earlane Sellers
University of Illinois
at Urbana-Champaign

Lynn Wolfmeyer
Western Illinois University

The third edition Task Force was expanded in 1990 to create a Computer Science section of the Articulation Guide. Those additional members were:

Rodney Angotti
Northern Illinois University

David Ballew
Western Illinois University

Jim Brunner
Black Hawk College

N. George Friedman, Jr.
University of Illinois
at Urbana-Champaign

Barbara Gentry
Parkland College

Ilga Higbee
Black Hawk College

Nick Mavros
Elgin Community College

The fourth edition of the Articulation Guide was prepared in 1994-95. The Illinois Articulation Initiative, created in January 1993, produced a lower-division General Education Core Curriculum that was adopted by the Illinois Board of Higher Education and Illinois Community College Board in September 1994. The adoption of this General Education Core Curriculum was a catalyst for creating the fourth edition Task Force whose charge was to update the Articulation Guide to be consistent with the course descriptions of the General Education Core Curriculum. Only the portion of the Articulation Guide that was affected by the Illinois Articulation Initiative was revised in the fourth edition. General Course Objectives were added to the revised course

descriptions to assist mathematics departments in meeting requirements mandated by the North Central Accreditation Association.

The fourth edition task force members were:

John Bradburn
Elgin Community College

David Clydesdale
Sauk Valley Community College

Neale Fadden
Belleville Area College

Rita Fischbach
Illinois Central College

James Hajek
Lincoln Land Community College

Dominic Magno
Harper College

Laverne McFadden
Parkland College

Jess Moore
John A. Logan College

Shirley Scheiner
Harry S Truman College

Stan Trail
Northern Illinois University

Diane Terlep
McHenry County College

The fifth edition of the Articulation Guide was prepared in 2003-2005. The Task Force was charged with reviewing and updating the Pretransfer (Developmental) Courses. The Task Force also developed guidelines for two combination courses – Combined Basic and Intermediate Algebra and Intermediate Algebra with Geometry. Only the portion of the Articulation Guide that dealt with the Pretransfer (Developmental) Courses was affected by the work of this Task Force.
The fifth edition task force members were:

Keven Hansen, Co-Chair
Southwestern Illinois College

Jim Harris, Co-Chair
John A. Logan College

Sandra Cox
Kaskaskia College

Richard Diefenbach
Shawnee Community College

Vern Kays
Richland Community College

Patricia Kiihne
Illinois College

Diane Koenig
Rock Valley College

Connie McLean
Black Hawk College

Bettie Truitt
Black Hawk College

The task force listed learning objectives for each course that was described as a means of helping colleges meet the standards for each course. Much discussion was given to aligning these courses with the AMATYC and/or NCTM standards; however it was decided that this guide was more content driven than pedagogically driven. The task force expressed that the manner of delivery is of utmost importance and each college, as well as each instructor, should consider using the Standards as a guide in teaching these courses.

The sixth edition of the Articulation Guide was prepared in 2006 – 2008. The Task Force was charged with reviewing and updating the Computer Science Courses and Recommended Courses of Study that the IMACC Curriculum Committee had recommended, approved, and asked that it be submitted for approval by a joint Task Force of ISMAA and IMACC. The subcommittee of the Curriculum Committee that worked on this task consisted of: Jim Harris, John A. Logan College; Bob Sompolski, Oakton Community College; Michael O'Leary, College of DuPage; Scott Reed, College of Lake County; and Jason James, Harper College. Only the portion of the Articulation Guide that dealt with the Computer Science was affected by the work of this Task Force.

The sixth edition task force members were:

Jim Harris, Co-Chair John A. Logan College	Paul McCombs, Co-Chair Rock Valley College
Bob Sompolski Oakton Community College	Keven Hansen Southwestern Illinois College
Mike O'Leary College of DuPage	Dan Hrozencik Chicago State University

The Task Force approved the listed learning objectives for each course described as a means of helping colleges meet the standards for each course. It is the desire of the Task Force that these descriptions and objectives be used as guidelines to enable students to receive a similar course regardless of place taken.

The order of presentation of topics in each course is neither meant to be given in the order of importance nor the order in which the topics should be presented in class. The guidelines are not meant to produce a rigid uniformity in courses throughout Illinois. The ICCB and the Joint Task Force recognize that the professional faculty member and college/university department shall make the judgment that best meets the needs of their students. The given minimum course content represents a consensus of Joint Task Force members, representing community college and four-year colleges. The framework still allows latitude in approach, emphasis, and choice of additional topics.

Students should plan their transfer program of study with a counselor/academic advisor and the catalog of the four-year college or university they plan to attend. The student is responsible for checking proper course selection with the senior institution.

As a general rule, courses taken at community colleges will not satisfy upper-division course requirements at senior colleges even though they may transfer as substitutes for upper-division courses.

The analytic geometry-calculus topics are relatively standard across the state universities and community colleges, but the sequencing of the topics may vary widely from institution to institution. Therefore, students are strongly advised to begin and complete the entire analytic geometry/calculus sequence at one institution.

Technology Statement

The appropriate use of technology is an essential part of many mathematics courses. Effective and strategic usage of technology by both students and faculty is highly encouraged. As is stated in the Crossroads in Mathematics: Standards for Introductory College Mathematics Before Calculus (AMATYC, 1995, p.12), Technology should be

used to enhance the study of mathematics but should not become the main focus of instruction. The amount of time that students spend learning how to use computers and calculators effectively must be compatible with the expected gain in learning mathematics. Computer software, especially packages appropriate for demonstration or visual representation of mathematical concepts, is strongly recommended. The use of calculators in any pre-algebra level course is best determined by departmental philosophy at the local level.

Standards Statement

In 1995 the American Mathematical Association of Two Year Colleges (AMATYC) published Crossroads in Mathematics: Standards for Introductory College Mathematics Before Calculus. The Illinois Mathematics Association of Community Colleges and the Mathematical Association of America are among the professional organizations that have reviewed and endorsed the philosophy and spirit of Crossroads in Mathematics. In the preface to Crossroads, Don Cohen, Editor, writes:

> This document is intended to stimulate faculty to reform introductory college mathematics before calculus. These standards are not meant to be the "final word." Rather they are a starting point for your actions.

The Joint Task Force of the Illinois Mathematics Association of Community Colleges (IMACC) and the Illinois Section of the Mathematical Association of America (ISMAA) is encouraged to use the Crossroads document as the starting point for their deliberations concerning possible modifications of the Illinois Mathematics and Computer Science Articulation Guide.

College Algebra Statement

While College Algebra and Precalculus courses are taught at post-secondary institutions where needed, these courses should not fulfill general education or quantitative literacy requirements. The content and instructional pedagogy applied in these courses should continue to be reviewed with the goal of preparing students to be successful in calculus and other courses that depend on a similar level of knowledge, rigor, and maturity. Adjustments to these courses should attempt to build upon appropriate changes in the K-12 curriculum that are a part of state-wide efforts to advance achievement for all students and, in particular, to smooth the transition from school to college.

Departments are advised not to attempt to design and teach college algebra and pre-calculus courses with the dual purpose as preparation for calculus and meeting goals for quantitative literacy and general education requirements. Expectations for mastery of the objectives considered essential preparation for subsequent calculus courses must take priority and time constraints, together with cognitive demands on the student group to be served, suggest such dual purpose courses are not likely to be successful.

I. Mathematics General Education Courses

I. Mathematics General Education Courses

General Education Requirements

In September 1994 the Illinois Board of Higher Education and the Illinois Community College Board adopted the General Education Core Curriculum developed by the Illinois Articulation Initiative. This General Education Core Curriculum is intended to be the model for the lower-division transfer General Education requirements on a statewide basis. The following is taken from the mathematics section of that document:

The mathematics component of general education focuses on quantitative reasoning to provide a base for developing a quantitatively literate college graduate. Every college graduate should be able to apply simple mathematical methods to the solution of real-world problems. A quantitatively literate college graduate should be able to:

- interpret mathematical models such as formulas, graphs, tables, and schematics, and draw inferences from them;
- represent mathematical information symbolically, visually, numerically, and verbally;
- use arithmetic, algebraic, geometric, and statistical methods to solve problems;
- estimate and check answers to mathematical problems in order to determine reasonableness, identify alternatives, and select optimal results; and
- recognize the limitations of mathematical and statistical models.

Courses accepted in fulfilling the general education mathematics requirement emphasize the development of the student's capability to do mathematical reasoning and problem solving in settings the college graduate may encounter in the future. General education mathematics courses should not lead simply to an appreciation of the place of mathematics in society, nor should they be merely mechanical or computational in character.

To accomplish this purpose, students should have at least one course at the lower-division level that emphasizes the foundations of quantitative literacy and, preferably, a second course that solidifies and deepens this foundation to enable the student to internalize these habits of thought.

1. *General Education Statistics*

3-4 semester hours

Prerequisites: Geometry and Intermediate Algebra both with a grade of C or better

Note: See Technology Statement in the Introduction.

Rationale

The general education statistics course provides students with an opportunity to acquire a reasonable level of statistical literacy and thus expand their base for understanding a variety of work-related, societal, and personal problems and statistical approaches to solutions of these problems. The main objective of the course is the development of statistical reasoning. Detailed techniques of statistical analysis and the mathematical development of statistical procedures are not emphasized. The course is intended to meet the general education requirement. It is not intended to be a prerequisite to nor a replacement for courses in statistical methods (for business or social science) nor for courses in mathematical statistics.

Major Areas and Topics

The following three major areas are to be considered. These, along with a listing of topics for each, are:

1. Organization, Presentation And Description Of Quantitative Data

    A. Graphical Methods
        1. Univariate Techniques - histograms, stem & leaf plots
        2. Bivariate Techniques - scatterplots (including estimation of best fit line)

    B. Numerical Methods
        1. Measures of Location - means, medians
        2. Measure of Variability - variances, standard deviations
        3. Measures of Association - correlation coefficients (properties, examples)

2. Probability And Probability Distributions

    A. Probabilities of events as relative frequencies from observed data.

    B. "Theoretical" probability as limit of relative frequencies.

        1. The "addition rule", conditional probability, "multiplication rule", independence of events.

    C. Sampling distributions of means and proportions (by simulation and/or example)

    D. Random variables, means and variances.

3. Sampling And Statistical Inference

   A. Population and Samples:
      1. random sampling, sample survey methods, errors in sampling, sampling distributions

   B. Point estimation: proportions, means and correlation coefs.; estimators and their properties.

   C. Interval estimation: proportions and means, standard errors, confidence intervals

   D. Hypothesis Testing
      1. testing H_0: $p = p_0$ and H_0: $\mu = \mu_0$ (using p-levels)

While some latitude in choice of topics and their position in the course is allowable, it is necessary that each of the major areas receive significant attention.

General Course Objectives

The student is expected to:

1. Organization, Presentation And Description Of Quantitative Data

   A. organize and graph univariate and bivariate quantitative data.

   B. know the definitions, properties, and functions of the following descriptive statistics and calculate their values from small data sets: means, medians, variances, standard deviations, correlation coefficients.

2. Probability And Probability Distributions

   A. view certain data sets as being the result of random experiments, determine the relative frequency of certain events related to these experiments and use probability language to express those determinations.

   B. express and provide examples of the interpretation of the probability of an event as the limit of the relative frequency of that event in repeated experiments, express and provide examples of alternative interpretations of probability.

   C. determine probabilities of events through the application of the standard ideas in elementary probability (e.g. the "addition rule," the "multiplication rule," counting techniques, independence of events, conditional probability,...).

   D. given a random experiment with a random variable defined on its sample space, write the probability function of the random variable and determine probabilities of events described in terms of random variables.

E. give examples of continuous random variables, their probability density functions, the determination of probabilities of events described in terms of random variables, and, for certain simple distributions (e.g. uniform or triangular), find probabilities of events, and the means and variance of the random variable.

F. construct and determine properties of sampling distributions of sample means and proportions.

3. Sampling And Statistical Inference

A. know and apply random sampling procedures and standard sample survey methods to develop sampling distributions of sample means and proportions.

B. state the Central Limit Theorem as it applies to sample means and proportions, know properties of normal and binomial distributions and find, by using tables, probabilities associated with random variables with these distributions.

C. know properties of estimators of population proportions and means and find corresponding estimates from sample data.

D. know properties of interval estimates of means and proportions and construct confidence intervals from sample data.

E. state appropriate hypotheses and alternatives concerning population means and proportions and test these using sample data.

2. *General Education Mathematics*

3-4 semester hours

Prerequisites: Geometry and Intermediate Algebra both with a grade of "C" or better.

Note: See Technology Statement in the Introduction.

This course is designed to fulfill general education requirements. It is not designed as a prerequisite for any other college mathematics course. This course focuses on mathematical reasoning and the solving of real-life problems. Three or four topics, chosen from the following list, are to be studied in depth. Mathematical modeling and/or projects is strongly recommended to be included as part of the course. The regular use of calculators and computers is strongly encouraged.

1. Counting techniques and probability
2. Game theory
3. Geometry (additional topics beyond the prerequisite)
4. Graph theory
5. Linear programming
6. Logic/Set theory
7. Mathematical modeling
8. Mathematics of finance
9. Statistics

Due to the diversity in the way the General Education Mathematics course can be designed, the objectives are general in nature and yet the learning outcomes must be specific to the topics chosen.

When designing this course, the specific learning outcomes for the topics selected must satisfy at least one of the general objectives listed below.

General Course Objectives

The student is expected to:

1. interpret mathematical models such as formulas, graphs, tables, and schematics, and draw inferences from them.
2. represent mathematical information symbolically, visually, numerically, and verbally.
3. use arithmetic, algebraic, geometric, and statistical methods to solve problems.

4. estimate and check answers to mathematical problems in order to determine reasonableness, identify alternatives, and select optimal results.

5. recognize the limitations of mathematical and statistical models.

3. *Quantitative Literacy*

3-4 semester hours

Prerequisite: Geometry and Intermediate Algebra with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This course is designed to provide the basic numeracy needed by a college graduate to reason quantitatively; that is, to reason about quantities, their magnitudes and their relationships between and among other quantities. This course is non-algorithmic in nature, rather conceptual understanding will be stressed. The course will not fulfill a mathematics requirement for the Bachelor of Science degree or for any science major in the Bachelor of Arts degree program.

In this course, students will develop competency in problem solving and analysis helpful to personal decision-making as well as to the decision-making needed by an educated citizen of the 21st century.

The activities listed below may be used to facilitate the desired problem solving, decision-making and quantitative reasoning competencies. Artificial problems should be avoided; the prerequisites should be solidly used. Hand-held calculators and personal computers should be used as tools in these activities.

1. Representing and analyzing data at that level of sophistication which considers the nature of such statistical measures as central tendency, dispersion, normal and chi square distributions, correlation and regression. Multi-step processes and decision-making based on hypothesis testing will be included. This section should encompass no more than one-third of the course.

2. Recognizing and using logical statements and arguments in a real world context.

3. Estimating, approximating and judging the reasonableness of answers.

4. Graphing and using polynomial functions and systems of equations and inequalities in the interpretation and solution of problems.

5. Selecting and using appropriate approaches and tools in formulating and solving real world problems from business and finance, from geometry and measurement, and from the environmental and biological sciences.

4. *Elementary Mathematical Modeling*

3-4 semester hours

Prerequisite: Geometry and Intermediate Algebra both with a grade of "C" or better.

Note: See Technology Statement in the Introduction.

Focuses on mathematical reasoning through the active participation of students in solving interesting and challenging problems. Integrates the use of graphing calculators and personal computers as problem solving tools. Emphasizes learning mathematics by doing mathematics so that students can build their own knowledge base of numeric, geometric, and algebraic models; and, at the same time, acquire the mathematical "habits of mind" necessary to use mathematics in their subsequent course work, their jobs, and their personal lives.

General Course Objectives

The student is expected to:

1. represent and solve problems using appropriate numeric, geometric, and algebraic models and state implied assumptions in modeling a problem solving situation.

2. use mathematically correct vocabulary and symbolism to communicate orally- and in writing-problem statements, problem-solving methods, and interpretations of the solutions to problems.

3. formulate a conjecture using inductive reasoning, support a conjecture using deductive reasoning, and refute a conjecture with a counter-example.

4. estimate solutions and perform order-of-magnitude comparisons to test the reasonableness of solutions or determine the best answer possible with the information available.

5. represent mathematical relationships using formulas, tables, and graphs.

6. solve problems by using graphing calculators or computers to create mathematical models.

Course Content

The core content of this course includes inductive and deductive reasoning in problem solving, mathematical proof, limitations of inductive and deductive reasoning, mathematical modeling as problem solving, and the limitations of mathematical models. Topics may include: sequences and series of modeling, variable and functions, graphical, tabular, and formulaic representation of algebraic functions, algebraic functions in modeling, logarithmic scales, logarithmic functions and exponential functions in modeling.

II. Mathematics
Pretransfer (Developmental) Courses

II. Mathematics Pretransfer (Developmental) Courses

In a study done in 1990 by the Conference Board of the Mathematical Sciences (Albers, Loftsgaarden, Rung, & Watkins, 1992), it was revealed that nearly 56% of the students studying mathematics in two-year college mathematics departments were studying at the remedial or developmental level. This translates to nearly 800,000 students taking developmental-remedial mathematics in 1995 (Loftsgaarden, Rung, & Watkins, 1995). In the decade since 1990, it would appear that this percent has not decreased, and more likely it has increased. Students continue to come to college ill-prepared in the area of mathematics, needing to take courses that will give them the background that they failed to obtain while in high school. Additionally, attrition rates in developmental-remedial mathematics range from 30% to 50% and are much higher in some urban and rural community colleges (Boylan, 1999; Illinois Community College Board, 2001; McCabe, 2000).

The purposes of developmental mathematics courses are:

1. to provide the foundation that students are lacking from their previous education.

2. to provide the student with the mathematical ability to function in their daily lives.

3. to enable the student to be successful in transfer level mathematics courses.

Developmental-remedial courses are crucial to the livelihood of most community college mathematics departments, but there continues to be much pressure from college stakeholders to get these students through the developmental-remedial courses and onto completion of their programs. The problem is that too few students are successfully completing these developmental-remedial courses (Batzer, 1997; DeMarois, 1998; Hashway, 1990; Illinois Board of Higher Education, 1997; Illinois Community College Board, 1997, 2001; McCabe, 2003). Hence a dilemma arises–prepare the students so they will be mathematically successful, but do it as quickly as possible.

In order for mathematics departments to address this dilemma, departments need to recognize that the students in these developmental-remedial courses often have a history of limited success in mathematics. Teaching these students using the traditional methods of their past is unlikely to be successful at the college level. It is important that instructors consider adapting their teaching methods to meet the needs of students. It is highly recommended that increased attention be given to:

- the active involvement of students in solving real multi-step mathematics problems.

- the introduction of needed skills in the context of real applications.

- mental arithmetic, estimation, and the translation of problem situations into algebraic models.

- the integration of mathematical topics so that students may use a wide range of mathematical content and techniques to solve problems.

- the conceptual understanding of mathematical ideas and the ability to use valid arguments.
- the appropriate use of technology throughout the curriculum for computational work, graphing, and geometry.
- the integration of interactive learning involving collaborative groups.
- the application of multiple approaches (numerical, graphical, symbolic, and verbal) to help the students learn a variety of techniques for solving problems (Cohen, 1995, p. 29).

As is stated specifically in the AMATYC Standards, basic reform of the content, intellectual development (expectation for student performance), and pedagogy is required. Problem solving and logical reasoning should be a main thread throughout all developmental-remedial courses. This does not imply the elimination of all skill development and algebraic manipulation that currently dominate the developmental-remedial curriculum; however, there needs to be adequate time for students to develop conceptual and mathematical understanding. Attention should be paid to these concerns as we design and teach the developmental-remedial courses.

References

Boylan, H. R. (1999). Exploring alternatives to remediation. Journal of Developmental Education, 22(3), 9.

Cohen, D. (Ed.). (1995). Crossroads in mathematics: Standards for introductory college mathematics before calculus. Memphis, TN: American Mathematical Association of Two-Year Colleges.

DeMarois, P. A. (1998). Facets and layers of function for college students in beginning algebra. Unpublished Ph.D., University of Warwick, Warwick, England.

Hashway, R. M. (Ed.). (1990). Handbook of developmental education. New York: Praeger.

Illinois Board of Higher Education. (1997). The scope and effectiveness of remedial/developmental education in Illinois public universities and community colleges (No. Item #8). Springfield, IL: Author.

Illinois Community College Board. (1997). Remedial/developmental education in the Illinois community college system (Report). Springfield, IL: Author.

Illinois Community College Board. (2001). Collaborating to strengthen student preparation (Task Force). Springfield, IL: Illinois Community College System.

Loftsgaarden, D. O., Rung, D. C., & Watkins, A. E. (1995). Statistical abstract of undergraduate programs in the mathematical sciences in the United States (MAA Report Number 2). Washington, D. C.: Mathematical Association of America.

McCabe, R. H. (2000). No one to waste: A report to public decision-makers and community college leaders. Washington, DC: Community College Press.

McCabe, R. H. (2003). Yes we can! A community college guide for developing America's underprepared. Phoenix, AZ: League for Innovation in the Community College.

1. *Arithmetic*

3–4 semester hours

Prerequisites: None

Note: See Technology Statement in the Introduction

This course is designed as a review of basic computational skills including operations with fractions, decimals, real numbers, percent, ratio and proportion, English and metric measurement, and formulas for area, perimeter and volume. Although emphasis should be placed on techniques and manipulations, problem solving and logical reasoning should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic and graphical representations of the concepts being taught wherever possible.

Course Content

1. Operations with whole numbers
2. Operations with fractions
3. Operations with decimals
4. Ratios and proportions
5. Percent and uses of percent
6. English and metric systems of measurement
7. Basic terms and formulas of geometry
8. Operations involving positive and negative real numbers
9. Solve application problems using the above content

Learning Objectives

1. Perform arithmetic operations with real numbers–whole numbers, integers, fractions, decimals and signed numbers.
2. Calculate and/or solve percentages, ratios, and proportions.
3. Convert within and use the English and metric measurement systems.
4. Use basic geometric terminology and formulas, such as perimeter, area and volume.
5. Use the above topics in routine applications.
6. Recognize the reasonableness of solutions.

II. Pre-Algebra

3–5 semester hours

Prerequisite: General knowledge of arithmetic

Note: See Technology Statement in the Introduction

This course is designed as a review of the basic operations of arithmetic and an introduction to algebra. This course should be a transitional course from a course that involves only arithmetic operations to the first course in Algebra. Although emphasis should be placed on techniques and manipulations, problem solving and logical reasoning should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic and graphical representations of the concepts being taught wherever possible.

Course Content

1. Integers and order of operations.
2. Solving linear equations and inequalities and applications, including formulas.
3. Operations with fractions and mixed numbers and solving equations containing fractions.
4. Operations with decimals and solving equations involving decimals.
5. Ratios and proportions with applications.
6. Solving percent problems including sales tax, commission, discount and interest.
7. Graphing linear equations and interpretation of graphs.
8. Basic geometric terminology and formulas involving perimeter, area, volume and measurement, using the English and metric systems.
9. Operations with polynomials and an introduction to factoring polynomials.
10. Introduction to square roots and applications of the Pythagorean Theorem.

Learning Objectives

1. Perform arithmetic operations with integers, rational numbers (fractional, decimal, and mixed number forms), real numbers, algebraic expressions and polynomials.
2. Solve linear equations and inequalities in one variable.
3. Solve and graph linear equations in two variables.

4. Apply the laws of exponents.
5. Apply geometric concepts of perimeter, area, and volume.
6. Demonstrate the basic concepts of roots and applications of roots.
7. Find the least common multiple using the prime factorization method.
8. Convert between fractional, decimal, and percent forms and apply these concepts to basic percent problems.
9. Apply the order of operations to numerical and algebraic expressions.
10. Apply the Pythagorean Theorem.
11. Use algebra to solve applications.
12. Solve real world problems involving measurement, percent, fractions, decimals and square roots.

3. *Basic Algebra*

3–5 semester hours

Prerequisites: Appropriate placement

Note: See Technology Statement in the Introduction

This course is designed to be a first course in Algebra. Although emphasis should be placed on techniques and manipulations, problem solving and logical reasoning should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic and graphical representations of the concepts being taught wherever possible.

Course Content

1. Review of arithmetic operations.
2. Review the properties of real numbers.
3. Graphing and solving linear equations and inequalities.
4. Applications of linear equations and inequalities, including formulas.
5. Solving systems of linear equations.
6. Introduction to factoring techniques and solving quadratic equations by factoring.
7. Operations with polynomials.
8. *Introduction to functions and function notation.
9. *Operations with rational expressions and solving rational equations.
10. *Operations with roots and radical expressions and solving radical equations.

Learning Objectives

1. Use the terms, definitions, and notation of basic algebra.
2. Identify and make use of real number properties and evaluate real number expressions.
3. Sketch the graph of a linear function and identify slope and intercepts.
4. Perform operations with polynomials.
5. Solve linear and quadratic equations.

6. Solve application problems and then recognize the reasonableness of solutions.

7. Apply laws of exponents.

8. *Demonstrate operations with rational expressions and solve rational equations.

9. *Perform basic operations with radical expressions and solve radical equations.

*Optional topics depending on the number of semester hours available

4. *Geometry*

2-4 semester hours

Prerequisite: Basic Algebra with a grade of "C" or better or appropriate placement

Note: See Technology Statement in the Introduction.

This course is designed to cover the fundamental concepts of geometry and is intended for students who lack credit in one year of high school geometry or need a review of the subject matter. Although emphasis should be placed on techniques and manipulations, problem solving and logical reasoning should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic, and graphical representations of the concepts being taught wherever possible. Deductive reasoning should be an integral part of the course. Algebraic concepts will be used where appropriate.

Course Content

1. Basic concepts of undefined terms, definitions, postulates, theorems, angles, and constructions.
2. The use of inductive reasoning and the writing of deductive (including indirect) proofs.
3. Congruent triangles.
4. Properties of parallel and perpendicular lines.
5. Parallelograms, regular polygons, and other polygons.
6. Ratios, proportions, and similarity .
7. Right triangles and applications of the Pythagorean Theorem.
8. Circles.
9. Concepts of perimeter, area, and volume.
10. *Geometric transformations.
11. *Concepts of locus in 2 and 3 space.

Learning Objectives

1. Use the concepts of undefined terms, definitions, postulates, and theorems in the logical development of geometry.
2. Perform constructions using a straightedge and compass and/or computer generated constructions.

3. Use inductive reasoning to form conjectures.
4. Write proofs using deductive (including indirect) reasoning.
5. Apply theorems of congruency to prove triangles and parts of triangles congruent.
6. Solve applications related to parallel and perpendicular lines.
7. Solve applications related to parallelograms, regular polygons, and other polygons.
8. Use similarity to solve applications.
9. Use the Pythagorean Theorem to solve applications.
10. Solve applications involving circles.
11. Apply formulas to solve problems related to perimeter, area, and volume.
12. *Apply the concepts of transformations.
13. *Apply the concept of locus.

*Optional topics depending on number of semester hours available

5. *Intermediate Algebra*

4–5 semester hours

Prerequisite: Basic Algebra and Geometry both with a "C" or better or appropriate placement

Note: See Technology Statement in the Introduction

This course is designed to be a second course in Algebra. Students must earn a grade of "C" or better in order to progress to transfer-level mathematics courses. Although emphasis should be placed on techniques and manipulations, problem solving and logical reasoning should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic and graphical representations of the concepts being taught wherever possible. The appropriate use of technology, such as a graphing calculator, is strongly encouraged.

Course Content

1. Solve linear equations and inequalities including absolute value equations and inequalities.
2. Graph linear and non-linear equations, including applications.
3. Introduction to functions, identifying range and domain, and graphing functions, including linear, quadratic, and absolute value.
4. Write equations of lines.
5. Operations with polynomials, factoring polynomials, solving quadratic equations and applications.
6. Solve systems of linear equations and applications in two and three variables.
7. Operations involving rational expressions; solving rational equations and applications.
8. Simplification and operations of radical expressions; solving radical equations and applications.
9. Introduction to complex numbers and elementary operations involving complex numbers.
10. Solve quadratic equations and inequalities, including rational inequalities.
11. *Introduction to exponential and logarithmic functions; solving and modeling applications.

Learning Objectives

1. Perform arithmetic operations with real numbers, complex numbers, and algebraic expressions including polynomials, rational expressions, and radical expressions.
2. Solve linear, rational, radical, absolute value, *logarithmic and *exponential equations in one and two variables with application of domain and range.
3. Solve linear inequalities and compound inequalities in one and two variables.
4. Factor polynomials, including binomials and trinomials, and identify prime polynomials.
5. Use various methods to solve quadratic equations, including the quadratic formula.
6. Write equations of lines and determine if lines are parallel or perpendicular.
7. Use graphs to identify solutions to linear equations and inequalities in one and two variables, as well as systems of equations and inequalities in two variables.
8. Solve systems of linear equations in two and three variables.
9. *Apply laws of logarithms and exponents to simplify logarithmic and exponential expressions and to solve equations and applications.
10. Graph quadratic, *exponential, and *logarithmic functions.
11. Solve applications involving linear expressions, equations and inequalities, rational equations, radical equations, and systems of equations.
12. Identify and solve applications involving direct, inverse and/or joint variation.

*Optional topics depending on number of semester hours available

5AG. Intermediate Algebra with Geometry

5-6 semester hours

Prerequisites: Basic Algebra with a grade of "B" or better or appropriate placement

Note: See Technology Statement in the Introduction.

This course is designed to be a combination of intermediate algebra and the fundamental concepts of geometry for those students who lack a second year of algebra and one year of high school geometry. It is also intended for those students who may need a review of the subject matter. Although emphasis should be placed on techniques and manipulations, problem solving, deductive proof writing, and logical thinking should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic and graphical representations of the concepts being taught wherever possible. Integration of algebraic and geometric topics should be a priority in this course. This course is appropriate for students who have been very successful in the prerequisite course or who received a strong placement score.

Course Content

1. Solve quadratic equations and inequalities, problem solving, and solving formulas for a specified variable.
2. Absolute value equations and inequalities.
3. Introduction to functions, identifying range and domain, and graphing functions, including linear, quadratic, and absolute value.
4. Solve systems of linear equations in two and three variables.
5. Operations with polynomials, factoring polynomials, solving quadratic equations and applications.
6. Operations with rational expressions; solving rational equations and applications.
7. Simplification and operations of radical expressions; solving radical equations and applications.
8. Introduction to complex numbers and elementary operations involving complex numbers.
9. *Introduction to exponential and logarithmic functions; solving and modeling applications.
10. Basic concepts of undefined terms, definitions, postulates, theorems, angles, and constructions.

11. The use of inductive reasoning and the writing of deductive (including indirect) proofs.
12. Congruent triangles.
13. Properties of parallel and perpendicular lines.
14. Parallelograms, regular polygons, and other polygons.
15. Ratios, proportions, and similarity.
16. Right triangles and applications of the Pythagorean Theorem.
17. Circles.
18. Concepts of perimeter, area, and volume.
19. *Geometric transformations.
20. *Concepts of locus in 2 and 3 space.

Learning Objectives

1. Perform arithmetic operations with real numbers, complex numbers, and algebraic expressions including polynomials, rational expressions, and radical expressions.
2. Solve linear, rational, radical, absolute value, *logarithmic and *exponential equations in one and two variables with application of domain and range.
3. Solve linear inequalities and compound inequalities in one and two variables.
4. Solve systems of linear equations in two and three variables.
5. Solve applications involving linear expressions, equations and inequalities, rational equations, radical equations, and systems of equations.
6. Use undefined terms, definitions, postulates, and theorems in the logical development of geometry.
7. Perform constructions using a straightedge and compass and/or computer generated constructions.
8. Use inductive reasoning to form conjectures.
9. Write proofs using deductive (including indirect) proofs.
10. Apply theorems of congruency to prove triangles and parts of triangles congruent.
11. Solve applications related to parallel and perpendicular lines.

12. Solve applications related to parallelograms, regular polygons, and other polygons.

13. Use similarity to solve applications.

14. Use the Pythagorean Theorem to solve applications.

15. Apply formulas to solve problems related to perimeter, area, and volume.

16 *Apply the concepts of transformations

17. *Apply the concept of locus.

*Optional topics depending on number of semester hours available

5BI. Combined Basic and Intermediate Algebra

5-6 semester hours

Prerequisite: "B" or better in the prerequisite course or appropriate Placement

Note: See Technology Statement in the Introduction

This course is designed to be a combination of basic and intermediate algebra. Students must earn a grade of "C" or better in order to progress to transfer-level mathematics courses. Although emphasis should be placed on techniques and manipulations, problem solving and logical reasoning should be a main thread throughout the course. Much effort should be given to utilize instruction that will provide students with needed techniques and also enable students to reason and make the connections that are involved in the learning of mathematics. The instruction should emphasize the connections between verbal, numerical, symbolic and graphical representations of the concepts being taught wherever possible. The appropriate use of technology, such as a graphing calculator, is strongly encouraged. This course is appropriate for students who have been very successful in the prerequisite course or who received a strong placement score.

Course Content

1. Review arithmetic operations.
2. Review the properties of real numbers.
3. Solve linear equations and inequalities including absolute value equations and inequalities.
4. Graph linear and non-linear equations, including applications.
5. Introduction to functions, identifying range and domain, and graphing functions, including linear, quadratic, and absolute value.
6. Write equations of lines.
7. Operations with polynomials, factoring polynomials, solving quadratic equations and applications.
8. Solve systems of linear equations and applications in two and three variables.
9. Operations involving rational expressions; solving rational equations and applications.
10. Simplification and operations of radical expressions; solving radical equations and applications.
11. Introduction to complex numbers and elementary operations involving complex numbers.

12. Solve quadratic equations and inequalities, including rational inequalities.

13. *Introduction to exponential and logarithmic functions; solving and modeling applications.

Learning Objectives

1. Use the terms, definitions, and notation of basic algebra.

2. Perform arithmetic operations with real numbers, complex numbers, and algebraic expressions including polynomials, rational expressions, and radical expressions

3. Solve linear, rational, radical, absolute value, *logarithmic and *exponential equations in one and two variables with application of domain and range.

4. Solve linear inequalities and compound inequalities in one and two variables.

5. Factor polynomials, including binomials and trinomials, and identify prime polynomials.

6. Use various methods to solve quadratic equations, including the quadratic formula.

7. Write equations of lines and determine if lines are parallel or perpendicular.

8. Use graphs to identify solutions to linear equations and inequalities in one and two variables, as well as systems of equations and inequalities in two variables.

9. Solve systems of linear equations in two and three variables.

10. *Apply laws of logarithms and exponents to simplify logarithmic and exponential expressions and to solve equations and applications.

11. Graph quadratic, *exponential, and *logarithmic functions.

12. Solve applications involving linear expressions, equations and inequalities, rational equations, radical equations, and systems of equations.

13. Identify and solve applications involving direct, inverse and/or joint variation.

*Optional topics depending on number of semester hours available

III. Mathematics Courses for Mathematics, Engineering, Computer Science and Business Majors

III. Mathematics Courses for Mathematics, Engineering, Computer Science,and Business Majors

1. College Algebra

4-5 semester hours

Prerequisites: Geometry and Intermediate Algebra both with a grade of "C" or better

Note: See Technology Statement in the Introduction.

The specified topics are considered a standard for the course. It is recommended that at least one of the further topics be included.

1. Specified Topics

   A. Minimal review of algebraic skills

   B. Functions and graphs

      1. Aspects of graphs
         a. Intercepts
         b. Symmetries
         c. Translations
         d. Reflections

      2. Aspects of functions
         a. Definition
         b. Domain
         c. Range
         d. Inverses
         e. Graphs

      3. Specific functions to be studied
         a. Polynomial
         b. Rational
         c. Exponential
         d. Logarithmic

   C. Conic sections

   D. Solution of systems

      1. Linear equations using matrices and determinants

      2. Non-linear equations and inequalities

   E. Theory of equations

      1. Synthetic division

      2. Factor and root theorems

3. Sequences, series, and binomial expansion

F. Further Topics

A. Permutations and combinations

B. Probability

C. Mathematical induction

General Course Objectives

The student is expected to:

1. demonstrate an understanding of the concepts related to functions and their inverses.

2. identify and graph quadratic, polynomial, rational, exponential, and logarithmic functions as well as the conic sections; also, demonstrate knowledge of the properties of these functions and relations and apply this knowledge to real-world situations.

3. demonstrate proficiency in solving linear and non-linear systems using various algebraic, matrix, and graphical methods.

4. graphically represent the solutions to inequalities and systems of inequalities that involve two variables.

5. use appropriate theorems and techniques to locate the roots of second and higher degree polynomial equations.

6. use the notation and formulae associated with arithmetic and geometric sequences and series.

7. demonstrate knowledge of binomial expansion, Pascal's triangle, and combinatorial formulae.

8. use technology appropriately in problem solving and in exploring and developing mathematical concepts.

2. *Trigonometry*

2-3 semester hours

Prerequisites: Geometry and Intermediate Algebra, both with a grade of "C" or better

Note: See Technology Statement in the Introduction.

The specified topics are considered as a standard for the course.

1. Specified Topics
   - A. Definitions, properties and graphical characteristics of trigonometric functions
   - B. Radian measure
   - C. Trigonometric identities and equations
   - D. Solution of oblique and right triangles
   - E. Inverse trigonometric functions
   - F. Powers and roots of complex numbers
2. Further Topics
   - A. Polar coordinates
   - B. Vectors

3. *College Algebra And Trigonometry*

5-6 semester hours

Prerequisites: Geometry and Intermediate Algebra, both with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This is an integrated course covering the topics from the College Algebra course and the Trigonometry course.

4. *Elementary Functions (Precalculus)*

3-6 semester hours

Prerequisites: Geometry and Intermediate Algebra, both with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This course would emphasize the notion of a function as a unifying concept for the topics of college algebra and an extension of the topics of trigonometry.

5. *Analytic Geometry-Calculus*

12-14 semester hours

Prerequisite: College Algebra and Trigonometry (separately or combined) with grade(s) of "C" or better; or Elementary Functions with a grade of "C" or better

Note: See Technology Statement in the Introduction.

The student must be strongly advised to complete this sequence at one institution. Transfer in the middle of the sequence is not recommended. If linear algebra is integrated into the sequence, the range of hours could be extended by two hours. If computer programming (with lab) is integrated into the sequence, the range of hours could be extended by one hour. This is a standard list of topics; it does not imply any particular order of topics.

1. Analytic Geometry
   - A. Coordinate systems
   - B. Lines and line segments; distance between points
   - C. Curve sketching
   - D. Equations and graphs of conic sections
   - E. Transformation of coordinates; translations and rotations
   - F. Parametric equations
   - G. Polar coordinates and equations
   - H. Vectors in 2 and 3 dimensions; vector operations
   - I. Planes and lines in space
   - J. Surfaces; quadric surfaces
   - K. Cylindrical and spherical coordinates
   - L. Space curves (optional)
2. Calculus
   - A. Limits and continuity
   - B. Definition of derivative; rate of change, slope
   - C. Derivatives of polynomial and rational functions
   - D. The chain rule
   - E. Implicit differentials
   - F. Approximation by differentials

G. Higher order derivatives

H. Rolle's theorem; mean-value theorem

I. Applications of the derivative

J. Anti-derivative

K. The definite integral

L. The fundamental theorem of calculus

M. Area, volume, other applications of the integral

N. The calculus of the trigonometric functions

O. Logarithmic and exponential functions

P. Techniques of integrations, including numerical methods

Q. Indeterminate forms; L'Hospital's rule

R. Improper integrals

S. Sequences and series; convergence tests; Taylor series

T. Functions of more than one variable; partial derivatives

U. The differential; directional derivatives; gradients

V. Double and triple integrals; evaluation and applications

6. *Differential Equations*

3-4 semester hours

Prerequisite: Appropriate Calculus Course with a grade "C" or better

Note: See Technology Statement in the Introduction.

The specified topics are considered a standard for the course. It is recommended that at least two or three of the further topics be included.

1. Specified Topics
   - A. Linear equations of first order
   - B. Linear equations with constant coefficients
   - C. The general linear equation
   - D. Variation of parameters
   - E. Undetermined coefficients
   - F. Linear independence; the Wronskian
   - G. Exact Equations
   - H. Separation of variables
   - I. Applications
2. Further Topics
   - A. Systems of linear differential equations
   - B. Solution by Laplace transforms
   - C. Existence and uniqueness of solutions
   - D. Solution by power series
   - E. Oscillation and comparison theorems
   - F. Partial differential equations
   - G. Boundary value problems
   - H. Numerical methods
   - I. Stability of solutions

7. *Introduction to Linear Algebra*

3 semester hours

Prerequisite: Appropriate Calculus Course with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This is a first course in vectors, matrices, vector spaces, and linear transformations. The ideas discussed in this course not only serve as an introduction to the more abstract courses a mathematics student meets at the junior-senior level, but also have many useful applications outside of mathematics. Since the topics listed below can be treated at many levels, great care must be taken to choose a text and methods of presentations that are appropriate for the second-year college students. The course is not intended to replace a more complete linear algebra course at the junior-senior level. This course could follow or be taken concurrently with the last course in the calculus sequence, but it should not replace the last calculus course for a transfer student.

1. Specified Topics
   - A. Vectors
   - B. Operations on matrices
   - C. Matrices
   - D. Inverse of a matrix
   - E. Solution of systems of linear equations
   - F. Rank of a matrix
   - G. Vector spaces and subspaces
   - H. Linear dependence and independence
   - I. Basis and dimension
   - J. Linear transformations
   - K. Sums, composites, inverses of linear transformations
   - L. Range and kernel of a linear transformation
2. Further Topics
   - A. Determinants
   - B. Eigenvalues and eigenvectors
   - C. Orthagonality; inner product spaces
   - D. Quadratic forms

8. *Mathematical Statistics*

3 semester hours

Prerequisite: Appropriate Calculus Course with a grade of "C" or better

Note: See Technology Statement in the Introduction.

There is wide variation in the statistics and probability courses in the four-year universities within the state. It is suggested that this course as taught at a community college contain topics chosen from the CUPM Mathematics 7 (probability and statistics--6 semester hours), that reflect the topics in a similar course at the university to which most students will transfer.

9. *Statistics (for Business Majors)*

3-4 semester hours

Prerequisite: College Algebra with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This course is designed especially for student in those fields which require a knowledge of descriptive statistics. This course does not, as a rule, count toward a mathematics major or minor. With consideration of the student being served, the topics will generally be selected from the following list.

1. Descriptive Methods
    - A. Frequency distributions and graphing
    - B. Measures of location--mean, median, quartiles, percentiles
    - C. Measures of variation--variance, standard deviation
2. Basic Probability Theory
    - A. Sample spaces, counting, factorials
    - B. Combinations, permutations
    - C. Probability Laws
3. Probability Distributions
    - A. Normal distribution and normal curve
    - B. Binomial distribution and its relation to the normal distribution
    - C. Random samples and sampling techniques
        1. Distribution of sample means and variance
        2. Applications in field such as quality control

4 Statistical Inference

- A. Estimation
- B. Hypothesis Testing
- C. The t-test and chi-square test
- D. Errors

5. Correlation and Regression
    A. Coefficient of correlation
    B. Regression line; line of best fit
6. F-Test and Analysis of Variance

10. *Finite Mathematics (A and B) (for Business and Management)*

3-4 semester hours

Prerequisite: College Algebra with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This course (either A or B) is designed especially for students in areas such as business, economics, social science, and non-physical sciences. It does not count towards a major or minor in mathematics. The student who wishes to transfer this course should check the specific requirements at the senior institution. This course should emphasize the concepts and applications of mathematics rather than mathematical structures. Because of the duplication in the content in these two alternatives, full credit should not be given for both courses, and a student should not be required to take one if the other has been successfully completed.

A. Usually Called Finite Mathematics

The topics listed are usually found in this course. Applications are drawn primarily from economics, business, and non-physical sciences.

1. Vectors, matrices, and matrix algebra
2. Solving systems of linear equations by matrix methods
3. Systems of inequalities and linear programming
4. Simplex method
5. Other applications of matrices
6. Set theory, logic, and Boolean Algebra
7. Counting and probability theory
8. Stochastic processes
9. Game theory
10. Markov Chain methods
11. Mathematical modeling
12. Mathematics of finance

B. Usually Called Applied Linear Algebra

This is a service course and not a course in abstract linear algebra for math majors and minors. Formal proofs should be avoided. The topics should be developed by appealing to intuition, geometry, and applications. The topics listed are usually found in this course.

1. Matrix Algebra
2. Systems of linear equations and matrices
3. Determinants
4. Vectors in 2-space and 3-space
5. Vector spaces
6. Eigenvalues and eigenvectors

11. *Calculus for Business and Social Science*

4 semester hours

Prerequisite: College Algebra with a grade of "C" or better

Note: See Technology Statement in the Introduction.

This calculus course is designed specifically for students in business and the social sciences and does not count toward a major or minor in mathematics. It emphasizes applications of the basic concepts of the calculus rather than proofs. This course may be taken before or after the course in finite mathematics. The student who wishes to transfer this course should check the specific requirements at the senior institution.

1. Introductory Topics
   - A. Sets, functions, linear functions
   - B. More general functions and curve sketching
   - C. Exponential and logarithmic functions
   - D. Applications of functions and graphs
   - E. Mathematical modeling
2. Differential Calculus
   - A. Limits, definition of the derivative
   - B. Formulas for finding derivatives
   - C. Higher derivatives
   - D. Maxima and minima of functions of one variable
   - E. Functions of more than one variable
   - F. Partial derivatives
   - G. Maxima and minima of functions of more than one variable
   - H. Applications in business and economics
3. Integral Calculus
   - A. The definite integral and the indefinite integral
   - B. The fundamental theorem of integral calculus
   - C. The use of definite integrals to find areas
   - D. Methods of integration: substitution, parts, tables ?
   - E. Approximate integration

of some variety

12. *Mathematics for Elementary Teaching I, II*

3-4 semester hours each

Prerequisites: Geometry and Intermediate Algebra both with a grade of "C" or better.

Note: See Technology Statement in the Introduction.

This two-course sequence is designed to meet the requirements of the state certification of elementary teachers. Students should be strongly encouraged to successfully complete both classes at the same institution. The student who wishes to transfer this sequence should check the specific requirements at the senior institution.

These courses fulfill the general education requirement only for students with a declared major in elementary and/or special education.

These courses focus on mathematical reasoning and problem solving. Course pedagogy involves students as active participants in the learning process. The use of calculators and microcomputers is strongly recommended for problem solving.

With consideration of the students being served, the topics will generally be selected from the following list:

1. Whole numbers
2. Sets, functions, and logic
3. Integers
4. Number theory
5. Rational numbers
6. Irrational numbers and the real number system
7. Probability
8. Statistics
9. Non-metric geometry
10. Measurement

General Course Objectives

The student is expected to:

1. solve problems and analyze solutions of problems that require logic.
2. gain knowledge and understanding of the mathematical content that is taught in elementary schools.

3. develop an appreciation of and interest in the history, structure, and applications of mathematics, including the philosophical base upon which the discipline rests.

4. develop an in-depth understanding of the fundamental operations of the arithmetic of real numbers.

5. develop an understanding of the nature and structure of the real number system.

6. learn the basic concepts of elementary probability and statistics.

7. learn how to plan, perform, and interpret statistical experiments.

8. study and interpret graphs and prepare graphs to display information.

9. make measurement estimates and compute accurate measurements of area, volume, time, and other measurable quantities, with particular emphasis placed on the metric system of measurements. (Methods of introducing and integrating measurement activities into the curriculum are discussed.)

10. analyze articles from professional journals and publications in the field of elementary school mathematics.

13. Discrete Mathematics

3-4 semester hours

Prerequisite: College Algebra with a grade of "C" or better.

Note: See Technology Statement in the Introduction.

Introduction to analysis of finite collections and mathematical foundations of sequential machines, computer system design, data structures, and algorithms. Includes sets and logic, subscripts, arrays, number systems, counting, recursion, graph theory, trees, nets, and Boolean algebra.

IV. Computer Science Courses and Recommended Courses of Study

1. *Foundations of Informational Technology (Computer Literacy)*

3–4 semester hours

Prerequisite: Intermediate Algebra and Geometry both with a "C" or better or appropriate placement

This course is designed to provide participants with a broad overview of computer concepts including key terminology and components of computer hardware, software, and operating systems and their use in problem solving.

Topics will include, but are not limited to, computer architecture, peripheral devices, networking components, system software, information system analysis, application software including word processing, database management, spreadsheet, and presentation software. Discussions will also include the Internet and web page development. This study of the uses and limitations of technology will lead to an informed decision about using computer resources.

This course is designed to fulfill general education requirements for a course in computing. It is not a prerequisite for any other college course and does not count as credit toward a degree in computer science. It does not meet the mathematics general education requirement.

Course Content

1. History of computers and how computers can be categorized.
2. Computer components, including input, output, storage, and communication devices as well as how these devices help process data.
3. System unit components including the motherboard, central processing unit, memory, ports, expansion slots, and buses.
4. System software including various operating systems, utilities, and programming languages.
5. Information system classification and factors in choosing a system.
6. Electronic commerce.
7. Application and productivity software such as home and personal, educational and reference, graphics and communication, including applications that focus on word processing, spreadsheets, presentation, image processing, and databases.
8. Telecommunication and networks involving networking components, data transmission characteristics, and communication media.
9. Applications of the Internet, viewing web pages, navigating the Internet and searching for information, as well as web page development.
10. Programming concepts and languages.
11. Multimedia and artificial intelligence.

12. Security issues and strategies including network and Internet risks as well as hardware and software risks.

13. Computer ethics.

14. Information Technology careers, opportunities, preparation, certificates, and work environments.

Learning Objectives

1. Understand, and use appropriately, common computer technology.

2. Summarize significant historical events in the development of computer technologies as well as the future outlook of computers.

3. List the various hardware components of a common computer system and define the function and common characteristics of each.

4. Explain the capabilities and limitations of a computer as a medium for representing, storing, manipulating, and communicating various forms of information.

5. Demonstrate an understanding of the computational capability of information processing software including spreadsheets, data bases, word processors, presentation software, and graphics software.

6. Demonstrate an understanding of operating system software and skill in the use of at least one type of operating system software to perform system tasks such as creating and deleting files and performing routine maintenance functions.

7. Demonstrate skill in the use of the Internet for retrieving information.

8. Demonstrate an understanding of the development of Web pages.

9. State the means by which a computer transforms information from human terms to digital form through a high level programming language.

10. Demonstrate the ability to analyze real-world problems and select appropriate computer resources to solve them.

11. Define a general network and compare various types of computer networks.

12. Specify how computers impact virtually every aspect of society and present at least one way computers impact on each student's chosen career field.

13. Demonstrate an understanding of security issues and strategies.

14. Explain computer ethics and ethical guidelines as well as privacy and property protection.

2. *Computer Science I*

3–4 semester hours

Prerequisite: Intermediate Algebra and Geometry with a grade of "C" or better.

This course is designed to be the first course for those who wish to study computer science, including students who are majoring in computer science, mathematics, or engineering. Topics include the history and ethics of computer science, software life cycle, debugging, data types, variables, decision statements, loops, arrays, functions, input/output, data abstraction, and objects.

Course Content

1. A brief introduction to the history of computer science starting with the first calculating machines to the beginning of electronic computing and modern technology.
2. Professional and public issues regarding use of computers including intellectual property.
3. Basic computer hardware including the central processing unit, read-only memory, random-access memory, input/output devices, and peripherals.
4. The different types of computer software: operating systems, high- and low-level languages, compilers, interpreters, clients, and servers.
5. The software life-cycle: problem definition, algorithm design, desktop testing, translation to a computer language, debugging.
6. Appropriate indenting, comments, variable names, and function names.
7. Basic output to the console and input from the keyboard.
8. Types, variable declarations, and object instantiation.
9. Writing expressions using variables, constants, operators, object methods, and object properties.
10. Program flow control with conditionals, loops, and selection statements.
11. Using libraries including strings and mathematical functions.
12. *Principles of graphical user interfaces; using GUI toolkits for input and output.
13. Program organization with functions and procedures including the use of parameters.
14. Introduction to recursion.
15. Introduction to object-oriented programming using data abstraction: inheritance, encapsulation, polymorphism.

16. Single and two-dimensional arrays.
17. Input and output to text files.

Learning Objectives

1. Enumerate several milestones in the history of computing.
2. Describe the professional and ethical obligations of computer programmers.
3. Describe the components of a personal computer.
4. Differentiate between the different types of computer software.
5. Organize a programming problem by specifying the required task, developing routines, checking the routines with paper and pencil, translating it to code, and then debugging.
6. Write programs that are readable and easily maintained.
7. Create programs that send output to the console and read input from the keyboard.
8. Declare variables using predefined types.
9. Perform mathematical and logical operations on variables by writing expressions.
10. Use program control statements including if-then-else structures and loops.
11. Manipulate strings (for example, insert, concatenation, and substrings) using a standard string object.
12. Write expressions using mathematical functions from a standard library.
13. *Building a simple graphical user interface for input and output.
14. Create functions and procedures and pass values to them using parameters.
15. Use local variables within functions and procedures.
16. Identify the recursive and base cases of a method using recursion and trace its execution.
17. Create user-defined objects.
18. Encapsulate an object's data and code so that other parts of the program will have only certain types of access to the object.
19. Use name overloading to create methods with different signatures.
20. Declare array variables and use them to display and manipulate lists of data.
21. Use a text stream to write to/read from disk.

*This is optional material for courses using the C++ language.

3. *Computer Science II*

3–4 semester hours

Prerequisite: Computer Science I with a grade of "C" or better.
College Algebra with a grade of "C" or better.

This course is designed to be the second course for those who wish to study computer science, including students who are majoring in computer science, mathematics, or engineering. Topics include object-oriented programming, classes, introduction to algorithms including basic searches and sorts, an introduction to dynamic data structures such as lists, stacks, queues and trees, event-driven programming with graphics, complexity analysis and recursion, records, and tables. Projects will be of a larger scale than in Computer Science I.

This course should use the same programming language as Computer Science I.

Course Content

1. Life-cycle of software: specification, design, risk analysis, verification, coding, testing, refining, production, and maintenance.
2. Modular program design using abstraction, object-oriented design, top-down design, design patterns, and modeling tools.
3. Appropriate program style that will simplify the task of the programmer and enhance the user's experience.
4. Implementation of small-sized to mid-sized projects.
5. Inheritance, class hierarchies, abstract classes, container classes, and iterators.
6. Polymorphism including examples of static binding and dynamic binding.
7. Fundamentals of event-driven programming including exception handling.
8. *Graphics basics, graphics APIs, and graphical user interfaces (GUI).
9. Program verification by rigorous proof.
10. Elementary complexity analysis with big-O metrics.
11. Introduction to algorithms using basic searches and sorts, recursion, and text processing.
12. Survey of linear (lists, stacks, and queues) and non-linear (binary search trees and heaps) data structures.
13. List variations: header nodes, doubly linked lists, and circular lists.
14. Traversal, insertion, and deletion algorithms in lists, trees, and heaps.

15. Memory issues including dynamically and statically allocated memory, external versus internal, and memory leaks.
16. Records and tables are introduced as means of organizing data.
17. Storing structured data on disk for later retrieval.

Learning Objectives

1. Organize a programming problem by specifying the required task, developing routines, checking the routines with paper and pencil, translating it to code, and then debugging.
2. Write programs that are robust, easy to modify, read, maintain, and update.
3. Design abstract data types using inheritance, encapsulation, and polymorphism to solve complicated problems.
4. Use class hierarchies to identify the inherited methods and properties of classes.
5. Create a container class with its own API.
6. Use an iterator to access to contents of a container class.
7. Create code that will respond to user events including appropriate exception handling.
8. *Use a standard graphical API to generate images to a canvas.
9. Use formal methods to verify the correctness of an algorithm including the use of mathematical induction.
10. Use big-O metrics to analyze elementary searching and sorting (selection, insertion, merge, quick, and heap) algorithms.
11. Program dynamic data handling routines using lists, stacks, queues, and trees.
12. Design an abstract data type that represents a linked list where methods are used to add, insert, delete, and display nodes.
13. Trace traversal, insertion, and deletion algorithms for linked list variations.
14. Use a binary search tree to quickly search and sort data.
15. Write recursive algorithms to traverse binary search trees and other simple tasks.
16. Use records to organize real world data sets.
17. Design tables to store information for later retrieval.

*This is optional material for courses using the C++ language.

4. *Computer Science III*

3–4 semester hours

Prerequisite: Computer Science II with a grade of "C" or better.
Discrete Mathematics with a grade of "C" or better.

This course is designed to be the third course for those who wish to study computer science, including students who are majoring in computer science, mathematics, or engineering. Topics include algorithms and algorithmic analysis, the design and implementation of data structures, hash tables, balanced binary search trees, graphs, recurrence relations, program complexity and efficiency, random number generation, and distributed methods. Projects will be of a larger scale than Computer Science II and will be team-based.

Course Content

1. Implementation of mid-sized projects using teams of programmers.
2. Survey of various sorting algorithms (insertion, shell, merge, heap, quick, and radix).
3. Review of basic data structures: lists, stacks, queues, trees, and heaps.
4. Tree variations: AVL trees and B-trees.
5. Advanced ADTs (sets, graphs, priority queues, and hash tables).
6. Hash functions and collision resolution.
7. Graph spanning tree algorithms, Kruskal's, and Dijkstra's algorithms.
8. Implementation strategies for determining which data structure is best for a given application.
9. Algorithmis paradigms (divide and conquer, greedy, dynamic, and back-tracking).
10. Use of recursion to implement backtracking, depth-first, and breadth-first searching algorithms.
11. Designing algorithms with regards to the efficiency of time and memory.
12. Complexity analysis using big-O, big omega, big theta, and little-O.
13. Use of recursion in complexity.
14. Advanced sorting algorithms that combine internal and external techniques based upon memory needs.
15. Random number generation.
16. Message passing and distributed algorithms.

17. Domain decomposition and granularity.

18. Measuring the efficiency of distributed algorithms including speedup and overhead.

Learning Objectives

1. Build medium-sized projects in teams of programmers.

2. Use a binary search tree to quickly search and sort data.

3. Create a hash table and write an efficient hash function.

4. Implement various types of graphs, search them, and write them to disk.

5. Use Cayley's formula or Kirchhoff's Theorem to find the number of spanning trees.

6. Implement algorithms for find minimum spanning trees.

7. Find the minimal path between two vertices of a weighted graph.

8. Given a particular set of data, choose the best data type to represent the collection.

9. Create functions that can be called recursively to solve problems including constraint satisfaction problems and recursion on graphs.

10. Describe the complexity and efficiency of algorithms using big-O, big omega, big theta, and little-O metrics.

11. Solve recurrence relationships as a result of algorithmic analysis.

12. Implement an external sorting routine for large amounts of data.

13. Analyze a random number generator to determine its period of pseudo-random numbers.

14. Decompose an algorithm into independent processes that minimize communication needs.

15. Implement a message passing algorithm to concurrently communicate data using semaphores and barriers.

16. Analyze the execution time, speedups, and overhead of various distributed algorithms.

5. *Computer Programming for Science and Engineering*

3–4 semester hours

Prerequisite: Calculus I with a grade of "C" or better

This course is designed to be a first course in programming of numerically intensive algorithms. Beyond a first course in computer programming, it should include a survey of introductory numerical methods. Topics include input, output and calculation of elementary data types, repetition and selection control structures, subprograms, and file and array processing. C++ or Java versions of the course should include class construction and object implementation. The addition of a computer algebra software tool can serve to complement the programming exercises with a mixture of symbolic and numeric enhancements.

Electrical and computer engineering generally require C, C++ or Java programming while other specialties prefer FORTRAN. The computer algebra material can be implemented in any of Mathematica, Maple, Derive or various Texas Instruments products (TI-89, TI-200, TI-Interactive).

Course Content

1. Programming language
   a. System software for editing, compiling, and executing programs.
   b. Hardware overview and how it is related to data representation.
   c. Debugging techniques in response to syntax and algorithmic errors.
   d. Input and output commands for numeric (FORTRAN integer, real, double precision; C/C++/Java int, float, double) and character (FORTRAN character; C/C++/Java char) data types.
   e. Arithmetic operations and assignment statements.
   f. Choosing one of multiple options with selection statements.
   g. Repeated execution of blocks of code.
   h. Modularity with subprograms (FORTRAN functions, subroutines; C/C++ functions, methods; Java methods).
   i. Intermediate data types (FORTRAN array, file; C array, file, struct; C++/Java array, file, class).
   j. Introduction to object-oriented programming using data abstraction: inheritance, encapsulation, polymorphism. (C++/Java)

2. Numerical methods
   a. Absolute and relative error analysis.
   b. Root searching algorithms and their limitations (Binary search and Newton methods).
   c. Derivative approximations at boundary and interior points (Forward, Central, and Backward approximations).
   d. Quadrature methods to approximate definite integrals (Trapezoid and Simpson methods).
   e. Integration methods to approximate differential equations (Euler and Runge-Kutte methods).
   f. Linear regression applied to discrete data.

3. Computer algebra system
   a. Symbolic and approximate calculations from algebra and calculus.
   b. Visualization techniques applied to continuous and discrete data.
   c. List operations including vector and matrix operations.
   d. Regression libraries applied to discrete data.

Learning Objectives

1. Programming language & numerical methods
   a. Compose, edit, and execute interactive programs performing input, output, and arithmetic operations on elementary numeric data types.
   b. Design programs demonstrating control statements that perform selection and repetition on blocks of code.
   c. Decompose algorithms into smaller segments using subprograms that communicate with parameters.
   d. Use sequential access files for input and output of batch programs.
   e. Demonstrate the manner in which arrays can be used to perform vector and matrix operations.
   f. Use error analysis to ascertain accuracy and reliability of numerical results.
   g. Apply iterative algorithms to search for roots of continuous functions.
   h. Design accumulation algorithms that demonstrate methods of quadrature, integration, and/or linear regression.
   i. Use files for input and output of regression points and/or matrix elements.
   j. Create user-defined objects.
   k. Encapsulate an object's data and code so that other parts of the program will have only certain types of access to the object.

2. Computer algebra system
   a. Understand the use and limitations of exact and numeric calculations as applied to algebra, calculus, and matrix operations.
   b. Visualize continuous functions and systems of equations.
   c. Use list operations to manipulate discrete data.
   d. Invoke regression library commands to model discrete cartesian points.
   e. Visualize discrete data points along with regression models.

Note: If this course is to be used by an Engineering Computer Science Major as the first course of the normal sequence of Computer Science I-II, this course needs to be in the same language as Computer Science II.

6. *Discrete Structures*

3–4 semester hours

Prerequisite: College Algebra with a grade of "C" or better.

Refer to the description of Discrete Mathematics (page 47) of this guide.

7. *Event Driven Programming*

3–4 semester hours

Prerequisite: Computer Science I with a grade of "C" or better

This course surveys event driven programming methods possessing Graphic User Interface (GUI) components that utilize Human Computer Interaction (HCI) screens for input. The intention is to keep this programming paradigm consistent, rather than in competition, with object-oriented paradigms. Additional topics include exception handling and an introduction to database manipulation.

The underlying assumption here is that the course is implemented in Java although other options such as Visual Basic and Javascript are available. The prerequisite allows the course to rapidly move through rudimentary programming practices to support the need to cover intermediate GUI construction and introductory database topics in the language.

Course Content

1. Review of intermediate constructs from CS I (files, arrays, classes).
2. Modes of execution: batch versus interactive and applications appropriate for each mode.
3. Methods of User Interaction (UI): command-line versus event-driven.
4. Event models, event source, and event listener.
5. Event handlers and event propagation.
6. Exceptions and their handling; multiple exception processing; exception hierarchies.
7. HCI styles and techniques.
8. Aspects of screen design: layout, color, fonts, and labeling.
9. Responsibilities of UI versus the application.
10. Language tools available for building a GUI; geometry management; mouse event monitors.
11. Widgets to aid in the processing of input/output: buttons, boxes, labels, text, and windows.
12. Network security management: authentication protocols, digital signatures, and digital certificates.
13. Introduction to Structured Query Language (SQL): select, insert, delete, and update queries.
14. Database objects and connectivity techniques.

Learning Objectives

1. Continue the development of object-oriented programs to produce maintainable applications.
2. Explain the difference between event-driven programming and command-line programming.
3. Understand the role that event handlers play between event sources and event targets.
4. Develop code that responds to exception conditions raised during execution.
5. Choose screen aspects to aid in HCI and techniques.
6. Utilize language layout managers to arrange screen fields.
7. Explain good design principles of widgets; sequenced screen presentations; and simple error-trap dialog.
8. Design, code, test, and debug simple event-driven programs that respond to user events such as completed input fields or mouse activity.
9. Describe methods used to create a secure connection to a data source.
10. Use basic SQL commands within an application.
11. Create applications that display, insert, modify, and delete data from a database.

8. *Computer Organization and Architecture*

3–4 semester hours

Prerequisite: Computer Science I with a grade of "C" or better

This course is a survey of the various levels of hierarchical computer architecture and design. The analysis of internal and external memory models, busses, I/O peripherals, complex instruction set computer (CISC), and reduced instruction set computer (RISC) processor strategies are covered. Additional topics include the instruction formats and addressing schemes of microprocessors such as Intel Pentium and Power PC architectures, vectorizing multiprocessors, and multi-computer systems.

Rather than a course in assembler language programming, rudimentary programming assignments in the language can be used to demonstrate aspects of computer architecture. Multi-computer and multiprocessor programming assignments can be demonstrated using environments such as MPI.

Course Content

1. A historical overview on how computer architectures have evolved including economic trends that have motivated the need for underlying technologies.

2. Computer performance measured in popular metrics: MIPS, MFLOPS, GFLOPS, and TFLOPS.

3. Integer data representation in various bases (binary, octal, and hexadecimal) and the means of performing arithmetic operations in the binary system.

4. Floating point numeric representation including normalization and IEEE standards.

5. The use of gates to represent elementary arithmetic and logical operations.

6. The program instruction cycle with fetch, execute, and interrupt activities.

7. Interconnection of components with busses demonstrating typical configurations (ISA, PCI) and bandwidths.

8. Hierarchical memory systems and their impact on performance, the principle of locality, caches, main memory, and virtual and external memory (disk, tape) strategies.

9. Input/Output (I/O) performance measures, types and characteristics of devices including the connections of I/O devices to processor, memory, and operating systems.

10. Arithmetic logic units, their construction, operations, and operands.

11. Instruction set design, operations in the instruction set, type and size of operands, and instruction representation.

12. Pipelining instruction data paths, pipelined control, addressing hazards of branching and dependencies, and exploiting more instruction-level parallelism.

13. Taxonomies of symmetric multiprocessors and distributed memory multi-computers, tightly coupled non-uniformed memory access systems, loosely coupled clusters, and memory access and cache coherence strategies.

Learning Objectives

1. Review the historical development of computers and computer architectures.

2. Apply the basic metrics by which new and existing computer systems may be evaluated to systems used in the course.

3. Understand information representation, error detection/correction schemes, and digital logic.

4. Describe elementary computer arithmetic operations for integer and floating point data in binary.

5. Identify the basic components of computer organization and understand how they work together.

6. Survey the hierarchical internal and external memory organization strategies and understand their impact upon performance.

7. Become knowledgeable about the design of I/O modules, control units, and arithmetic logic units.

8. Learn the format of instruction sets, addressing modes, and the operation of the instruction cycle.

9. Understand the relative strengths and weaknesses of CISC and RISC architectures.

10. Demonstrate the use of pipelining and vector processing including data path and dependencies.

11. Recognize current superscalar microprocessor and multiprocessor models in today's market.

9. *Computer Science Major*

Recommended Course of Study: Associate of Science
Major in Computer Science

For the student ready for Calculus	
1. Freshman 1	**3. Sophomore 1**
A. Computer Science I B. Calculus I C. Discrete Mathematics	A. Computer Science III B. Computer Organization and Architecture
2. Freshman 2	**4. Sophomore 2**
A. Computer Science II B. Calculus II	A. Event Driven Programming

For the student not ready for calculus	
1. Freshman 1	**3. Sophomore 1**
A. Computer Science I B. Precalculus	A. Computer Science III B. Computer Organization and Architecture C. Calculus I
2. Freshman 2	**4. Sophomore 2**
A. Computer Science II B. Discrete Mathematics	A. Event Driven Programming B. Calculus II

Note: The specific curriculum of study should be planned with appropriate advisors in accordance with specific requirements of the university to which students wish to transfer.

A student entering this program of study not adequately prepared to begin with the suggested courses should plan to enroll in the correct prerequisite courses and should plan to enroll in additional semesters in order to complete the expectations of this curriculum.

10. *Engineering Computer Science Major*

Recommended course of study: Associate of Science

1. Freshman 1	3. Sophomore 1
A. Calculus I B. Discrete Mathematics	A. Computer Science II B. Calculus III C. Computer Organization & Architecture
2. Freshman 2	**4. Sophomore 2**
A. Computer Programming for Science and Engineering B. Calculus II	A. Computer Science III B. Linear Algebra and/or C. Differential Equations

Note: The specific curriculum of study should be planned with appropriate advisors in accordance with specific requirements of the university to which students wish to transfer.

A student entering this program of study not adequately prepared to begin with the suggested courses should plan to enroll in the correct prerequisite courses and should plan to enroll in additional semesters in order to complete the expectations of this curriculum.

11. *Mathematics, Physical Science, or Engineering (Mechanical, Industrial, Electrical) Major*

Recommended course of study: Associate of Science

1. Freshman 1	3. Sophomore 1
A. Calculus I	A. Computer Organization and Architecture B. Calculus III
2. Freshman 2	**4. Sophomore 2**
A. Computer Programming for Science and Engineering B. Calculus II	A. Linear Algebra and/or B. Differential Equations

Note: The specific curriculum of study should be planned with appropriate advisors in accordance with specific requirements of the university to which students wish to transfer.

A student entering this program of study not adequately prepared to begin with the suggested courses should plan to enroll in the correct prerequisite courses and should plan to enroll in additional semesters in order to complete the expectations of this curriculum.

12. *Business Curriculums*

Recommended course of study: Associate of Arts

1. Freshman 1	3. Sophomore 1
A. Foundations of Information Technology B. College Algebra	A. Computer Science II B. Finite Mathematics
2. Freshman 2	**4. Sophomore 2**
A. Computer Science I B. Business Calculus	A. Event Driven Programming B. Business Statistics

Note: The specific curriculum of study should be planned with appropriate advisors in accordance with specific requirements of the university to which students wish to transfer.

A student entering this program of study not adequately prepared to begin with the courses suggested as the entry points should take the appropriate prerequisite courses and should plan to enroll in additional semesters in order to complete the expectations of this curriculum.

V. Additional Course Options as Recommended by the Association of Computing Machinery (ACM)

Illinois Articulation Initiative (IAI) Panel on Computer Science has defined the content of Computer Science courses that will be acceptable for transfer credit for the State of Illinois; whereas The Association for Computing Machinery (ACM) has recommended course content on a national level. The IAI Panel referred to the recommendations of the Two-Year College Education Committee of ACM in developing their transfer guidelines. A complete exposition of IAI can be found at URL www.itransfer.org and a complete exposition of the ACM recommendations for Two-Year Computer Science programs can be found at the URL www.acmtyc.org

Each organization has recommended the content for the Computer Science I – Computer Science III courses, which is very similar other than the order and the manner topics are covered. As an attempt to inform IMACC members of the ACM recommendations and to generate further discussion on the issues of content, prerequisites, order of content, manner of delivery, and objectives for the Computer Science I – Computer Science III courses, the following course descriptions will offer a different route to accomplish the course content of the courses that are currently approved by the IAI Computer Science Major Panel. The Net-Centric Operating systems course description has been included to offer a new possibility for elective option.

It is important to note that the courses that follow are not approved by the IAI Computer Science Major Panel as equivalent or replacements for any of the IAI computer science courses; however, as the computer science courses change with time, these course descriptions may prove helpful as a guide.

Computer Science I – Version A

3–4 semester hours

Prerequisite: Intermediate Algebra and Geometry with a grade of "C" or better in each.

This course is designed to be the first course for those who wish to study computer science, including students who are majoring in computer science, mathematics, or engineering. Topics include the history and ethics of computer science, software life cycle, debugging, data types, variables, decision statements, loops, arrays, functions, input/output, data abstraction, and objects.

Course Content

1. A brief introduction to the history of computer science starting with the first calculating machines to the beginning of electronic computing and modern technology.
2. Professional and public issues regarding use of computers including intellectual property.
3. Basic computer hardware including the central processing unit, read-only memory, random-access memory, input/output devices, and peripherals.
4. The different types of computer software: operating systems, high- and low-level languages, compilers, interpreters, clients, and servers.
5. The software life-cycle: problem definition, algorithm design, desktop testing, translation to a computer language, debugging.
6. Appropriate indenting, comments, variable names, and function names.
7. Basic output to the console and input from the keyboard.
8. Types, variable declarations, and object instantiation.
9. Writing expressions using variables, constants, operators, object methods, and object properties.
10. Program flow control with conditionals, loops, and selection statements.
11. Program organization with functions and procedures including the use of parameters.
12. Single and two-dimensional arrays.
13. Using libraries including strings and mathematical functions.
14. Input and output to text files.
15. Introduction to object-oriented programming using data abstraction: inheritance, encapsulation, polymorphism.

Learning Objectives

1. Enumerate several milestones in the history of computing.
2. Discuss current trends in computer science.
3. Describe the professional obligations of computer programmers and the role of codes of conduction.
4. Identify the rights of copyright and patent holders and the consequences of infringement upon those rights.
5. List various types of computer crime and explain how the software can be designed to combat it.
6. Describe the components of a personal computer.
7. Differentiate between the different types of computer software.
8. Organize a programming problem by specifying the required task, developing routines, checking the routines with paper and pencil, translating it to code, and then debugging.
9. Write programs that are readable and easily maintained.
10. Create programs that send output to the console and read input from the user.
11. Instantiate common objects (such as strings), call their methods, and access their properties.
12. Declare variables using predefined types.
13. Perform mathematical and logical operations on variables by writing expressions.
14. Use program control statements including if-then-else structures and loops.
15. Create functions and procedures and pass values to them using parameters.
16. Use local variables within functions and procedures.
17. Declare array variables and use them to display and manipulate lists of data.
18. Use a text stream to write to/read from disk.
19. Manipulate strings (for example, insert, concatenation, substrings) using a standard string object.
20. Write expressions using mathematical functions from a standard library.
21. Create user-defined objects.
22. Define objects that inherit properties and methods from other objects.

23. Encapsulate an object's data and code so that other parts of the program will have only certain types of access to the object.

24. Use polymorphism to create objects that have a single interface for different but similar tasks.

Computer Science II – Version A

3–4 semester hours

Prerequisite: Computer Science I with a grade of "C" or better in each.

This course is designed to be the second course for those who wish to study computer science, including students who are majoring in computer science, mathematics, or engineering. Topics include object-oriented programming, classes, introduction to algorithms including basic searches and sorts, introduction to dynamic data structures such as stacks and queues, event-driven programming, graphics, virtual machines, records, tables, and memory issues. Projects will be of a larger scale than in Computer Science I.

This course should use the same programming language as Computer Science I.

Course Content

1. Life-cycle of software: specification, design, risk analysis, verification, coding, testing, refining, production, and maintenance.
2. Modular program design is achieved using abstraction, object-oriented design, top-down design, design patterns, and modeling tools.
3. Appropriate program style is emphasized that will simplify the task of the programmer and enhance the user's experience.
4. Implementation of small- to mid-sized projects.
5. Program verification.
6. Abstract data types: more advanced examples of inheritance, encapsulation, and polymorphism than found in Computer Science I.
7. Advanced class topics including class hierarchies; abstract, interface, and container classes; and iterators.
8. Introduction to algorithms using basic searches and sorts, recursion, and text processing.
9. Introduction to dynamic data structures such as sets, stacks, queues, and graphs.
10. Fundamentals of event-driven programming including exception handling.
11. *Graphics basics, graphics APIs, graphical user interfaces (GUI), color models, and the mathematics of graphics.
12. Virtual machines including the hierarchy of virtual machines and intermediate languages.
13. Memory issues including dynamically and statically allocated memory, external versus internal, and memory leaks.

14. Records and tables are introduced as means of organizing data.
15. Storing structured data on disk for later retrieval.

Learning Objectives

1. Organize a programming problem by specifying the required task, developing routines, checking the routines with paper and pencil, translating it to code, and then debugging.
2. Select an appropriate design and implement it in a programming solution.
3. Write programs that are easy to modify, read, maintain, and update.
4. Write programs that are easy to use, well-documented, and catch user errors.
5. Use formal methods to verify the correctness of an algorithm including the use of mathematical induction.
6. Design abstract data types using inheritance, encapsulation, and polymorphism to solve complicated problems.
7. Use class hierarchies to identify the inherited methods and properties of classes.
8. Create a container class with its own API.
9. Use an iterator to access to contents of a container class.
10. Use elementary searches (like the binary search) and sorts (such as the selection sort).
11. Write recursive algorithms to solve simple tasks.
12. Program dynamic data handling routines using stacks and queues.
13. Create code that will respond to user events including appropriate exception handling.
14. *Use a standard graphical API to generate images to a canvas.
15. Perform mathematical transformations to produce animation.
16. Explain what a virtual machine is and how it is used to make a program portable.
17. Compare the positive and negative aspects of compilation and interpretation.
18. Manage memory usage so that distributed solutions are able to execute on various machines.
19. When appropriate, write garbage collection routines that prevent the program from crashing the operating system.

20. Use records to organize real world data sets.
21. Design tables to store information for later retrieval.

*This is optional material for courses using the C++ language.

Computer Science III – Version A

3–4 semester hours

Prerequisite: Computer Science II with a grade of "C" or better.
Recommended: Discrete Mathematics with a grade of "C" or better.

This course is designed to be the third course for those who wish to study computer science, including students who are majoring in computer science, mathematics, or engineering. Topics include algorithms and algorithmic analysis, the design and implementation of data structures, heaps, hash tables, linked lists, graphs and trees, recursion, advanced searches and sorts, program complexity and efficiency, and random number generation. Projects will be of a larger scale than Computer Science II and will be team-based.

Course Content

1. Implementation of mid-sized projects using teams of programmers.
2. Designing algorithms with efficiency in mind.
3. Heap implementation including the delete-max, delete-min, decrease-key, insert, and merge operations.
4. Looking up information in hash tables.
5. Linked Lists: adding, inserting, deleting, and displaying nodes.
6. Trees and other types of graphs such as binary search trees, spanning trees, linear orders, and partial orders.
7. Implementation strategies for determining which data structure is best for a given application.
8. Algorithms involving graphs such as transversals, shortest-path algorithms, transitive closure, and minimal spanning trees.
9. In-depth study of recursion including backtracking and recursion on graphs.
10. Advanced sort and search algorithms.
11. Analyzing program complexity and determining best and worst case scenarios.
12. Measuring the efficiency of algorithms including execution times, algorithm growth rates, big-O and other notation.
13. Random number generation.

Learning Objectives

1. Build medium-sized projects in teams of programmers.
2. Work with heaps using the heap sort or selection algorithms.
3. Create a hash table and write an efficient hash function.
4. Design an abstract data type that represents a linked list where methods are used to add, insert, delete, and display nodes.
5. Implement various types of graphs, search them, and write them to disk.
6. Use a binary search tree to quickly search and sort data.
7. Use Cayley's formula or Kirchhoff's Theorem to find the number of spanning trees.
8. Analyze algorithms for find minimum spanning trees.
9. Given a particular set of data, choose the best data type to represent the collection.
10. For any partial order find an algorithm that will select every element exactly once.
11. Use algorithms to find a path between two vertices of a graph that has minimum weight.
12. Given an arbitrary relation, find the smallest transitive relation that contains it.
13. Create functions that can be called recursively to solve problems including constraint satisfaction problems and recursion on graphs.
14. Write search and sort routines that are the most efficient for a given problem.
15. Analyze the execution time of various algorithms.
16. Describe the complexity and efficiency of an algorithm using big-O, big omega, big theta, and little-O notation.
17. Analyze a random number generator to determine its period of pseudo-random numbers.

Net Centric Operating Systems

3–4 semester hours

Prerequisite: Computer Science II with a grade of "C" or better.

This course introduces the fundamentals of operating systems design and implementation. Topics include an overview of the components of an operating system, mutual exclusion and synchronization, implementation of processes, scheduling algorithms, memory management, and file systems. It introduces the structured, implementation, and the theoretical underpinnings of computer networking and the applications enabled by that technology.

This course could contain an integrated laboratory component using a recent version of Linux, Unix, or Microsoft Windows operating systems and the Java programming language. Emphasis should be placed on the built-in networking and security capabilities of modern operating systems. Simple and secure network socket scripting can be accomplished with built-in methods from the Java networking class.

Course Content

1. Concurrent execution; states and state diagrams, implementation structures (ready lists, process control blocks, etc.); dispatching and context switching; and interrupt handling.

2. Mutual exclusion; deadlock detection, prevention; solution strategies; models and mechanisms (semaphores, monitors, condition variables, and rendezvous); producer-consumer problems; synchronization; and multiprocessor issues.

3. Preemptive and non-preemptive scheduling; scheduling policies; processes and threads; and real-time issues.

4. Physical memory and memory management hardware; overlays, swapping, and partitions; paging and segmentation; page placement and replacement policies; working sets and thrashing; and caching.

5. Fundamental file system concepts (data, metadata, operations, organization, buffering, and sequential vs. nonsequential files); content and structure of directories; file system techniques (partitioning, mounting and unmounting, and virtual file systems); memory-mapped files; special-purpose file systems; naming, searching, and access; and backup strategies.

6. Overview of system security; policy/mechanism separation; security methods and devices; protection, access, and authentication; models of protection; memory protection; encryption; and recovery management.

7. Network standards and standardization bodies; the ISO 7-layer reference model in general and its instantiation in TCP/IP; circuit switching and packet switching; streams and datagrams; physical layer networking concepts; data link layer concepts; Internetworking and routing; and transport layer services.

8. Protocols at the web application layer; principles of web engineering web sites; remote procedure calls; lightweight distributed objects; the role of middleware; support tools; security issues in distributed object systems; and enterprise-wide web-based applications.

9. Issues of network management; issues for Internet Service Providers (ISPs); security issues and firewalls; and quality of service issues.

10. Basic data compression; audio compression and decompression; image compression and decompression; video compression and decompression; and performance issues.

11. Multimedia data technologies; multimedia standards; capacity planning and performance issues; input and output devices; MIDI keyboards, synthesizers; storage standards; multimedia servers and file systems; and tools to support multimedia development.

12. Introduction to LANs and WANs; layered protocol design, ISO/OSI, IEEE 802; impact of architectural issues on distributed algorithms; network computing; and distributed multimedia.

Learning Objectives

1. Understand concurrent execution and the synchronization mechanisms available to avoid deadlock or congestion control.

2. Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems, such as priority, performance comparison, and fair-share schemes.

3. Differentiate the mechanisms used in interfacing a range of devices (including hand-held devices, networks, and multimedia) to a computer.

4. Compare and contrast different approaches to file organization, including distributed file systems.

5. Recognize the current network standards and their impact upon switching and layer strategies

6. Understand the basic concepts of network security including authentication, integrity, key distribution, and system security design challenges.

7. Utilize Web applications to demonstrate an example of client-server computing.

8. Design scripts with middleware tools used in distributed operating systems such as DCOM or CORBA.

9. Summarize the principles of virtual memory, caching, paging, and segmentation.

10. Compare various compression and decompression algorithms for different data types, such as text, graphics, sound, and video.

11. Understand the impact that data compression has on network traffic.
12. Recognize the current architectures for networks and distributed systems.

Conclusion

The Joint Task Force thanks Ivan Lach and Yvonne Singley of the Illinois Community College Board staff for their input and assistance in the development of this guide. The Joint Task Force also thanks all our colleagues who provided assistance with the work. The contributions made by Mathematics Panel of the Illinois Articulation Initiative were appreciated in the writing of the fourth edition of this guide.

We recognize and thank the Illinois Community College Board and its professional staff for providing the leadership in continuing the effective articulation of mathematics courses among community colleges and senior colleges and universities in Illinois.

We recommend that the Illinois Community College Board staff continue to monitor the effectiveness of this guide and appoint another joint task force when articulation issues merit the next review.

Addendum Reflecting Dates of Revisions and Dates of Course Description Updates

Articulation Guide

1st edition	Called the Curriculum Guide	1969-1973
2nd edition	Called the Curriculum Guide	1981-1982
3rd edition	Expanded to create a Computer Science Section in 1990	1989-1991
4th edition	Dealt with General Education Core Curriculum Courses only	1994-1995
5th edition	Dealt with Pre-Transfer Developmental Courses only	2003-2005
6th edition	Dealt with Computer Science Courses only	2006-2008

Printings of the 4th Edition

Each of the printings dealt with some minor corrections and/or additions to the Articulation Guide that both IMACC and ISMAA approved at a meeting of their memberships. The dates that follow are the dates of the printings of the 4th Edition with those agreed to corrections and/or additions.

2nd printing	June, 1998
3rd printing	August, 2000
4th printing	May, 2002
5th printing	May, 2003

Printings of the 5th Edition

This printing dealt with some minor corrections and/or additions to the Articulation Guide that IMACC and ISMAA approved at a meeting of their memberships. The date that follows is the date of printings of the 5th edition with those agreed to corrections and/or additions or a reformatting of the current document.

2nd printing	September, 2006

Courses and Date of Revision

General Education

General Education Statistics	March, 1995
General Education Mathematics	April, 1998
Quantitative Literacy	April, 1998
Elementary Mathematical Modeling	August, 2000

Mathematics Pretransfer Courses

Arithmetic	April, 2005
Prealgebra	April, 2005
Beginning Algebra	April, 2005
Geometry	April, 2005
Intermediate Algebra	April, 2005
Intermediate Algebra with Geometry	April, 2005
Combined Basic and Intermediate Algebra	April, 2005

Mathematics Courses for Mathematics, Engineering, Computer Science, and Business Majors

College Algebra	March, 1990
Trigonometry	March, 1990
College Algebra and Trigonometry	March, 1990
Analytic Geometry – Calculus	March, 1990
Differential Equations	March, 1990
Introduction to Linear Algebra	March, 1990
Mathematical Statistics	March, 1990
Statistics (for Business Majors)	March, 1990
Finite Mathematics (A and B) (for Business Management)	March, 1990
Calculus for Business and Social Science	March, 1990
Mathematics for Elementary Teachers I, II	March, 1995
Discrete Mathematics	April, 1998

Computer Science Courses and Recommended Courses of Study

Foundations of Informational Technology (Computer Literacy)	April, 2008
Computer Science I	April, 2008
Computer Science II	April, 2008
Computer Science III	April, 2008
Computer Programming for Science and Engineering	April, 2008
Discrete Structures	April, 2008
Event Driven Programming	April, 2008
Computer Organization and Architecture	April, 2008
Computer Science I – Version A	April, 2008
Computer Science II – Version A	April, 2008
Computer Science III – Version A	April, 2008
Net Centric Operating Systems	April, 2008