

**The Faculty of the
Mathematics Division
Institute of Mathematical Sciences and Physics
College of Arts and Sciences
University of the Philippines Los Baños**

hereby accepts this Special Problem entitled

**Combinatorics Kit Plus:
A CAI on Combinatorics with an Authoring System Feature
and a Combinatorics Calculator**

by

Marc Christian Saribay

in partial fulfillment for the
requirements of the degree of

Bachelor of Science in MATHEMATICS

Prof. Durwin C. Santos
Adviser

Date Signed

Dr. Maria Cristeta N. Cuaresma
Head, Mathematics Division

Date Signed

Prof. Genaro A. Cuaresma
Director, IMSP

Date Signed

**Combinatorics Kit Plus:
A CAI on Combinatorics with an Authoring System Feature
and a Combinatorics Calculator**

A Special Problem
Presented to the Faculty of the
Institute of Mathematical Sciences and Physics
College of Arts and Sciences
University of the Philippines Los Baños

In Partial Fulfillment of the Requirements
For the Degree of

Bachelor of Science
in
MATHEMATICS

by

Marc Christian Saribay

October, 2004

ACKNOWLEDGMENT

There are so many people I want to thank and acknowledge for their support. First of all, I would like to thank my adviser **Prof. Durwin C. Santos** for dedicating his time and assistance as well as his willingness to support every step of my progress upon working on this SP. I would also like to acknowledge **Prof. Rolando G. Panopio** who presented me with the problem and who also assisted me on my progress. They inspired me to extend my capabilities while working on this SP.

I would like to thank all of my instructors. They really helped a lot for without them, I may not have been able to achieve my accomplishments.

I wish to extend special thanks to **Ramon K. Maristanes Jr., Michael Figueroa, Ronald Riveta, Daniel Ezra Añoso**, and **Lyonelle Bernardo** for the support and the good times. To **Katherine Abarabar, Elaine de Jesus, Ramon Antonio Castro** and **Jimson Serrano** who used to spend vacant hours with me but unfortunately, already left the campus, thanks for the good times. Thanks to all my blocmates (Bloc 38 math students of batch 2000).

Special thanks to my **STS 1 “Vibra” groupmates** namely **Princess, Chiqui, Sherryl, Aimee, Rona, Paul, Uris, and Rex** for the fun moments this semester despite the hardships. Once again for the good times, thanks to my PE 2 chess classmates **Aileen Aquino, Lorenzo Encinares, Marilyn Amoin**, and **Mary Joy Agsalud** and to my PI 100 classmates **Cenon Dinno Eborra** and **Raggie Ranald Agdon**, thanks!

I would like to thank **Danilo Briones Jr.** and **Noel Patrick Obrero** for the printing support, CD-R discs, and again... the good times. To my previous housemates **Harley Lacaden, Cohnel Serrano, Ian Martinez, Mike, Rey, and Russel**, just remember our great gimmicks way back then. Thanks to my high school classmates here in LB especially to **Michele Lopez, Jolline Natividad, Recah Austria, Roxanne Caraig** and many others. Some of them have already graduated. Interaction with these people is critical to the success of my work. To all of my classmates, I want to thank you all.

Above all, I want to thank God for giving me countless blessings and for giving me enough strength to face the hardships of life.

I dedicate this to my family.

Marc Christian Saribay

**Combinatorics Kit Plus:
A CAI on Combinatorics with an Authoring System Feature
and a Combinatorics Calculator**

Author: Marc Christian Saribay
Adviser: Prof. Durwin C. Santos

ABSTRACT

A CAI (Computer-Aided Instruction) authoring system specifically for Combinatorics was developed. The CAI in Combinatorics named *Combinatorics Tutor* would help a student learn Combinatorics through interactive lectures. The CAI could also evaluate a student's learning capacity by presenting some combinatorial problems that the student needs to solve. Moreover, a Combinatorics calculator named *Combinatorics Kit* which would supplement the CAI was included. These programs constitute the *Combinatorics Kit Plus*.

Program development was done using Microsoft Visual Basic 6. An appropriate database structure for the Combinatorics Tutor learning interface was created in Microsoft Access. For the CAI interface design, log in system was implemented so that every student could have their own profile. Lecture pages and exam items were created in a form of a web page. Randomization of exams was implemented so that different students could never get same exam items all over again. Tutor, the core program of Combinatorics Tutor, was designed to serve as a template for other fields of study. Tutor Database Manual was made for the instructions on the creation of CAI contents for Tutor.

A second release of the Combinatorics calculator named Combinatorics Kit was included to supplement the Combinatorics Tutor CAI. The user interface was redesigned and new combinatorial methods were added in this release.

TABLE OF CONTENTS

CHAPTER I: Introduction	1
A. Background of the Study	1
B. Statement of the Problem	2
C. Significance of the Study	3
D. Objectives of the Study	4
E. Limitations of the Study	5
F. Date and Place of Study	5
CHAPTER II: Review of Related Literature	6
CHAPTER III: Theoretical Framework	9
A. CAI (Computer-Aided Instruction)	9
B. Permutations and Combinations	12
C. Microsoft Visual Basic 6.0	13
CHAPTER IV: Methodology	14
CHAPTER V: Results and Discussion	16
A. Combinatorics Tutor	16
B. Tutor Database Manual	22
C. Combinatorics Kit	27
D. Combinatorics Kit Plus CD	41
CHAPTER VI: Summary and Conclusion	42
CHAPTER VII: Recommendations	44
CHAPTER VIII: References	45

CHAPTER I: INTRODUCTION

A. Background of the Study

In the field of Mathematics, different kinds of analysis are introduced to solve mathematical problems. Proper interpretation of the problem must be the first step in order to determine the approach that will give the solution to the problem. The next step will be the application of the correct methods in order to arrive at a particular solution. Counting, in its basic essence, is a basic knowledge in mathematics but in dealing with the possibilities of arrangement and ordering of objects, applying the basic knowledge in counting would be a complicated task. Combinatorics, a field of study in Mathematics which involves the analysis in counting and enumerating the number of possibilities of the arrangement and order of objects, was introduced to lessen the complexities of these counting problems.

It is hard to learn Combinatorics by simply reading books and understanding its concepts. A level of interaction while learning is essential and it is done by presenting a set of questions that a student needs to answer. This method evaluates the student's capacity on learning the lectures. This could be successfully done through the use of a CAI (Computer-Aided Instruction). Through the use of a computer, a CAI could help a student learn Combinatorics through interactive lectures. Moreover, a CAI could also evaluate a student's performance by presenting some combinatorial problems that the student needs to solve.

Furthermore, Combinatorics requires a lot of computations. There are many formulas and patterns that need to be studied, remembered, and analyzed. Sometimes, it takes a long time to solve for computations involving long sequences. Although scientific calculators offer aid on computations, punching the buttons of factorial and some nCr and nPr functions in a scientific calculator takes some series of steps. This approach would seem impractical in dealing with many values. Also, it will not show the needed answer right away.

With the aid of a computer, a CAI and a program which could handle combinatorial computations could be created but knowledge in computer programming is necessary. In creating a computer program, knowledge about the syntax of the programming language is a necessity before writing the code. The program flow and the programming structures to be used will depend on the approach that is used to solve the particular mathematical problem. The user interface of the program also matters. Since Windows is a dominant platform, a Windows-based combinatorial program would appeal readily to the users.

B. Statement of the Problem

This project intends to create a Windows-based program called the Combinatorics Kit Plus. The program will be designed to aid anyone taking up a combinatorics course or even beginners who want to learn Combinatorics. Combinatorics Kit Plus will be presented in two parts. The *Combinatorics Tutor* will present a CAI for learning Combinatorics. The CAI will contain lectures on

Combinatorics as well as quizzes and exams to evaluate the student's performance. It will be designed for the purpose of interactive Combinatorics learning. The other part would be the *Combinatorics Kit* which will offer a calculator for solving combinatorial problems. In an easy to use interface, it will be able to solve combinatorial problems by offering quick solutions using some implemented combinatorial methods. It will be designed primarily for the purpose of faster combinatorial computations.

C. Significance of the Study

The result of the project will be an easy to use CAI which will offer interactive Combinatorics learning and a Combinatorics calculator program which will offer some combinatorial methods as a tool to provide fast combinatorial computations. The CAI will be able to evaluate a student's performance through exams. On the other hand, the calculator program will also be able to generate a list of sequential values on some combinatorial methods like the Combination and the Stirling Number. Thus, the result will be useful to students studying Combinatorics and even beginners in a Combinatorics course as they could learn Combinatorics through the use of the CAI and they will have a quick way to know the answer to a computation using the Combinatorics calculator program. Moreover, the calculator program can be used by researchers who may want to readily check conjectures leading to theorems.

D. Objectives of the Study

The general objective of the study is to develop a Windows-based computer program called Combinatorics Kit Plus which will offer *Combinatorics Tutor* (CAI on Combinatorics) as the learning tool for Combinatorics and *Combinatorics Kit* (Combinatorics calculator) as an aid for faster combinatorial computations. The specific objectives of study are the following:

Combinatorics Tutor

1. To design the database structure appropriate for the learning interface
 2. To design a basic user interface that will run in the Windows platform
 3. To create a working computer program by writing the program code that would handle CAI events
 4. To create the student log-in system that will set individual student profiles on the database as well as the lecture, quizzes, and exam pages which will be integrated in the program through the database
 5. To determine the capabilities of the programming language in handling the randomization of exams
 6. To thoroughly test and debug the program for errors, and
 7. To package the program in a distributable media for deployment
-

Combinatorics Kit

1. To design the user interface of the program that will run in the Windows platform
2. To create a working computer program by writing the program code that would handle computer inputs
3. To determine the capabilities of the programming language in handling the combinatorial methods until it reaches its limitations
4. To thoroughly test and debug the program for errors, and
5. To package the program in a distributable media for deployment

E. Limitations of the Study

Combinatorics Kit Plus will only include the Permutations and Combinations part of the Combinatorics study. The lectures, quizzes and exams on Combinatorics Tutor will be based on this coverage. The combinatorial methods that will be present in Combinatorics Kit will also be limited to the said coverage. Moreover, Combinatorics Kit will be limited by the numeric data type precision handling of the programming language and the current technology and speed of computers.

F. Date and Place of Study

This study has been conducted from June to September, 2004 at the University of the Philippines Los Baños, College, Laguna.

CHAPTER II: REVIEW OF RELATED LITERATURE

Computer programmers have been able to create computer-aided instruction programs that have served to increase student learning by affecting cognitive processes and increasing motivation. Current research shows the mechanisms by which computer programs facilitate this learning: (1) personalizing information, (2) animating objects on the screen, (3) providing practice activities that incorporate challenges and curiosity, (4) providing a fantasy context and (5) providing a learner with choice over his/her own learning.

Numerous studies on the effectivity of computer-aided instruction have been conducted. Most of these studies have been done in United States.

Personalizing information allows computer-aided instruction to increase learner interest in the given tasks (Padma and Ross, 1987) and increase the internal logic and organization of the material (Anderson, 1984; Ausubel, 1968; Mayer, 1975; Rumelhart and Ortony, 1977). New information can be more easily integrated into existing schema if a student's name or other familiar contexts appear in a problem.

The animation of objects involved in the explanation of a particular concept, for example, Newton's First Law of Motion, increases learning by decreasing the cognitive load on the learner's memory thereby allowing the learner to perform

search and recognition processes and to make more informational relationships (Reiber, 1991).

Computer-assisted instruction increases motivation by providing a context for the learner that is challenging and stimulates curiosity (Malone, 1982). Activities that are intrinsically motivating also carry other significant advantages such as personal satisfaction, challenge, relevance, and promotion of a positive perspective on lifelong learning (Keller and Suzuki, 1988; Kinzie, 1990).

A fantasy context increases learning by facilitating engagement (Parker and Lepper, 1992; Malone, 1982). Fein (1981) and Signer (1987) have also found, apart from using computer programs, that involvement in fantasy is often highly intrinsically motivating.

Providing students with choice over their own learning provides learner-controlled instruction, which contributes to motivation. Increased motivation in turn increases student learning (Kinzie, Sullivan and Berdel, 1988). Also, program-controlled instruction, as opposed to learner-controlled, may get in the way of the learner by requiring the learner to study all of the given subject matter rather than only the elements the learner needs (Mayer, 1964). Further, learner-controlled instruction makes it possible for individuals to make certain choices in an activity and to affect certain outcomes. As a result, the individual feels competent and self-determining, and the activity has greater personal meaning and intrinsic interest

(DeCharms, 1968; Lepper, 1985). Tennyson (1980, 1981) found learner control to be instructionally effective when individuals were given advisement on their performance in relation to program criteria. Further, learner control results in more positive attitudes toward the instruction (Fry, 1972; Hurlock, Lahey, and McCann, 1974).

A number of studies on CAI have been conducted in the University of the Philippines Los Baños. Some of these works include “CAIMM: Computer Aided Instruction in Mathematical Modeling” by Osuna which presents a CAI in Mathematical Modeling developed in Flash 5; “ECONCAI: Computer Assisted Instruction for General Economics” by Roguel which presents a CAI in Economics; “CAIGRAPH: Computer-Aided Instruction for Computer Graphics” by Cataggutan; “A Computer-Aided Instruction Program for CMSC 11 Subject” by Juliano; and many others.

Authoring systems are the key to future expansion in CAI (Zettersten, 1985). Thus, this study contributes a CAI in Combinatorics with an authoring system feature and a Combinatorics calculator.

CHAPTER III: THEORETICAL FRAMEWORK

A. CAI (Computer-Aided Instruction)

Computer-Aided Instruction (CAI) is a diverse and rapidly expanding spectrum of computer technologies that assist the teaching and learning process. CAI is also known as computer-assisted instruction. Examples of CAI applications include guided drill and practice exercises, computer visualization of complex objects, and computer-facilitated communication between students and teachers.

Information that helps teach or encourages interaction can be presented on computers in the form of text or in multimedia formats, which include photographs, videos, animation, speech, and music. The guided drill is a computer program that poses questions to students, returns feedback, and selects additional questions based on the students' responses. Recent guided drill systems incorporate the principles of education in addition to subject matter knowledge into the computer program.

Computers can also help students visualize objects that are difficult or impossible to view. For example, computers can be used to display human anatomy, molecular structures, or complex geometrical objects. Exploration and manipulation of simulated environments can be accomplished with CAI—ranging from virtual laboratory experiments that may be too difficult, expensive, or

dangerous to perform in a school environment to complex virtual worlds like those used in airplane flight simulators.

CAI systems can be categorized based on who controls the progression of the lesson. Early systems were linear presentations of information and guided drill, and control was directed by the author of the software. In modern systems, and especially with visualization systems and simulated environments, control often rests with the student or with the instructor. This permits information to be reviewed or examined out of sequence. Related material also may be explored. In some group instructional activities, the lesson can progress according to the dynamics of the group.

CAI can dramatically increase a student's access to information. The program can adapt to the abilities and preferences of the individual student and increase the amount of personalized instruction a student receives. Many students benefit from the immediate responsiveness of computer interactions and appreciate the self-paced and private learning environment. Moreover, computer-learning experiences often engage the interest of students, motivating them to learn and increasing independence and personal responsibility for education.

Although it is difficult to assess the effectiveness of any educational system, numerous studies have reported that CAI is successful in raising examination scores, improving student attitudes, and lowering the amount of time required to

master certain material. While study results vary greatly, there is substantial evidence that CAI can enhance learning at all educational levels.

In some applications, especially those involving abstract reasoning and problem-solving processes, CAI has not been very effective. Critics claim that poorly designed CAI systems can dehumanize or regiment the educational experience and thereby diminish student interest and motivation. Other disadvantages of CAI stem from the difficulty and expense of implementing and maintaining the necessary computer systems. Some student failures can be traced to inadequate teacher training in CAI systems. Student training in the computer technology may be required as well, and this process can distract from the core educational process. Although much effort has been directed at developing CAI systems that are easy to use and incorporate expert knowledge of teaching and learning, such systems are still far from achieving their full potential.

A good CAI should deliver information in the most entertaining way possible to motivate the students to learn. Multimedia capabilities such as animations, pictures, videos, music, and narrations enhance the interest of students. The CAI must also conduct evaluation from time to time by presenting questions and returning feedback. A student controlled CAI is better considering that many students prefer the self-paced and private learning environment. In effect, the CAI is known to offer fast learning rate. It also improves the attitude of students.

B. Permutations and Combinations

Permutations and Combinations are certain arrangements of objects or elements. In the case of combinations, no attention is paid to the order of arrangement. In permutations, however, different orderings are counted as distinct, and repetitions of the elements selected may or may not be allowed.

Combinatorics is founded on the formulas for permutations and combinations. Combinatorics is involved in the analysis of counting and enumerating the number of possibilities of the arrangement and order of objects. The two basic counting principles are the following:

Addition Principle

- If the first task can be performed in m ways, while a second task can be performed in n ways, and the two tasks cannot be performed simultaneously, then performing either task can be accomplished in any one of $m + n$ ways.

Multiplication Principle

- If a procedure can be broken down into first and second stages, and if there are m possible outcomes for the first stage and n for the second stage, then the total procedure can be carried out, in the designated order, in mn ways.

Let $A = \{ a_1, a_2, \dots, a_n \}$ be given set of n distinct objects. For $0 \leq r \leq n$, an **r-permutation** of A is a way of arranging any r of the objects of A in a row. When $r = n$, an n -permutation of A is simply called a permutation of A .

$$P_r^n = n(n-1)(n-2)\dots(n-r+1) = \frac{n!}{(n-r)!}$$

Now, Let A be a set of n distinct objects. A **combination** of A is a subset of A . More precisely, for $0 \leq r \leq n$, an r -combination of A is an r -element subset of A .

$$C_r^n = \frac{P_n^r}{r!} = \frac{n!}{r!(n-r)!}$$

These formulas served as the foundation of all the other methods such as the Circular permutation, Pairing, Stirling numbers, and many others.

C. Microsoft Visual Basic 6.0

Microsoft Visual Basic 6.0 is the programming language that was used to create the interface of the Combinatorics CAI and the Combinatorics Calculator. Interface creation is the first step. The interface of the program could be easily designed by selecting the appropriate Windows controls (text boxes, command buttons, etc.) and drawing it on the blank window (form). The controls have a list of corresponding events (for example, a command button control has a mouse click event) and a code could be written to handle the particular event. Visual Basic 6 also provides several ways to connect to a database. Furthermore, it offers a control capable of displaying Web Page/HTML files.

Visual Basic 6 is a powerful, object-oriented, and event-driven development system designed for rapid development of Windows-based applications.

CHAPTER IV: METHODOLOGY

The study was conducted in six stages. Each stage was essential to the development of the software. The first stage was the research stage. It is the stage where information, basic knowledge, as well as important ideas and concepts on Combinatorics were gathered. References dealing with Permutations and Combinations and references on Visual Basic 6 offered necessary data for the study. Some developed CAI was studied for the development of a better CAI. The lessons and exams on the CAI were planned well. The database structure of the CAI authoring system was carefully thought of before starting with its user interface.

The next stage was the design stage. A windows-based program offers a graphical user interface (or GUI). The user interface of the program was easily designed in Visual Basic 6. The needed controls (text boxes, command buttons, etc.) were simply placed on the form (a form is represented by a blank window) for user interaction. The user interface of the Combinatorics calculator has been redesigned and was streamlined.

The third stage was the writing of the program code. A program code was written for the events which need to be handled. An example of an event is a mouse click. A piece of code would determine the action of the program. For example, when the user clicks a command button at runtime, the code written on the mouse click event of a command button would be analyzed and the interpreted action of the code will be triggered.

The fourth stage was the determination of limitations of the program. This stage was essential on the Combinatorics Calculator. Computers are limited by currently existing technology. Due to memory limitations, the length of a numeric variable could not be infinite. Thus, upon declaration of a numeric variable, the

programming language sets the fixed maximum length of the variable in the memory space. If a value to be stored on the variable exceeds the maximum length, an error will be generated by the programming language. The limit on the numeric precision upon declaration of a numeric variable would also be a limiting factor for the program to be created.

The fifth stage was the testing and debugging of the program. After the completion of the program, it went through series of tests. This was the most exhaustive part of the study. The program was thoroughly tested by starting the program and testing its functionality. Errors that were encountered during runtime were debugged. All possible sorts of input were considered to locate the errors and a message box was used to inform the user to correct the input. The program was also tested on a low-end system to evaluate its performance. If the program runs poorly on a low-end system, the program will be adjusted to give a better performance on a wide range of system configurations.

Finally, the program was packaged for deployment. After the program was thoroughly tested and the bugs were already fixed, it needs to be released and deployed. The program has been proven stable and already works in the desired way so it was compiled in an executable form to run independently on a Windows system. The distribution media that was used is a CD. This CD contains an installation wizard to install the program easily in Windows systems.

CHAPTER V: RESULTS AND DISCUSSION

Program development has been done using Microsoft Visual Basic 6.0 on a Pentium 4 – 1.5 Ghz computer running Microsoft Windows XP Professional. The **Combinatorics Kit Plus** is composed of two parts. The first part is the CAI with Combinatorics content called ***Combinatorics Tutor***. The second part is an improved version of the Combinatorics calculator called ***Combinatorics Kit***.

Combinatorics Tutor

Combinatorics Tutor serves as the CAI part of Combinatorics Kit Plus. At first, it was intended to simply provide Combinatorics content. Because the CAI has been designed to depend on the databases, it was made possible to have an authoring system feature on the CAI itself. Thus, Tutor was built and served as the front-end for the Combinatorics Tutor. Combinatorics Tutor and the Tutor (front-end) were intended to have different setup programs. With the inclusion of Combinatorics content, Combinatorics Tutor was intended to work in conjunction with the Combinatorics Kit.

Database Structure (Fig. 1) – The first thing that was carefully thought of was the database structure that will be appropriate for the learning interface. After finalizing the design of the database structure, the databases were created in Microsoft Access 2003 and were converted to Access 97 format to support Visual Basic 6. The databases include the student information and status as well as the page links for lectures and exams. Two Access databases were created. The first database file is Tutor.mdb. It contains four tables namely Lectures, Students, System, and Titles. Every table consists of several fields which serve as categories for a record on the table. This database file handles the student profiles, the titles of the parts of the study, and the organization of the lectures. On the other hand, System.mdb was designed for the exams and was intended to be duplicated for each user when a user tries to take an exam. Once this database file was duplicated, it would

be renamed accordingly and it would contain the randomized exam contents as its records for a certain user.

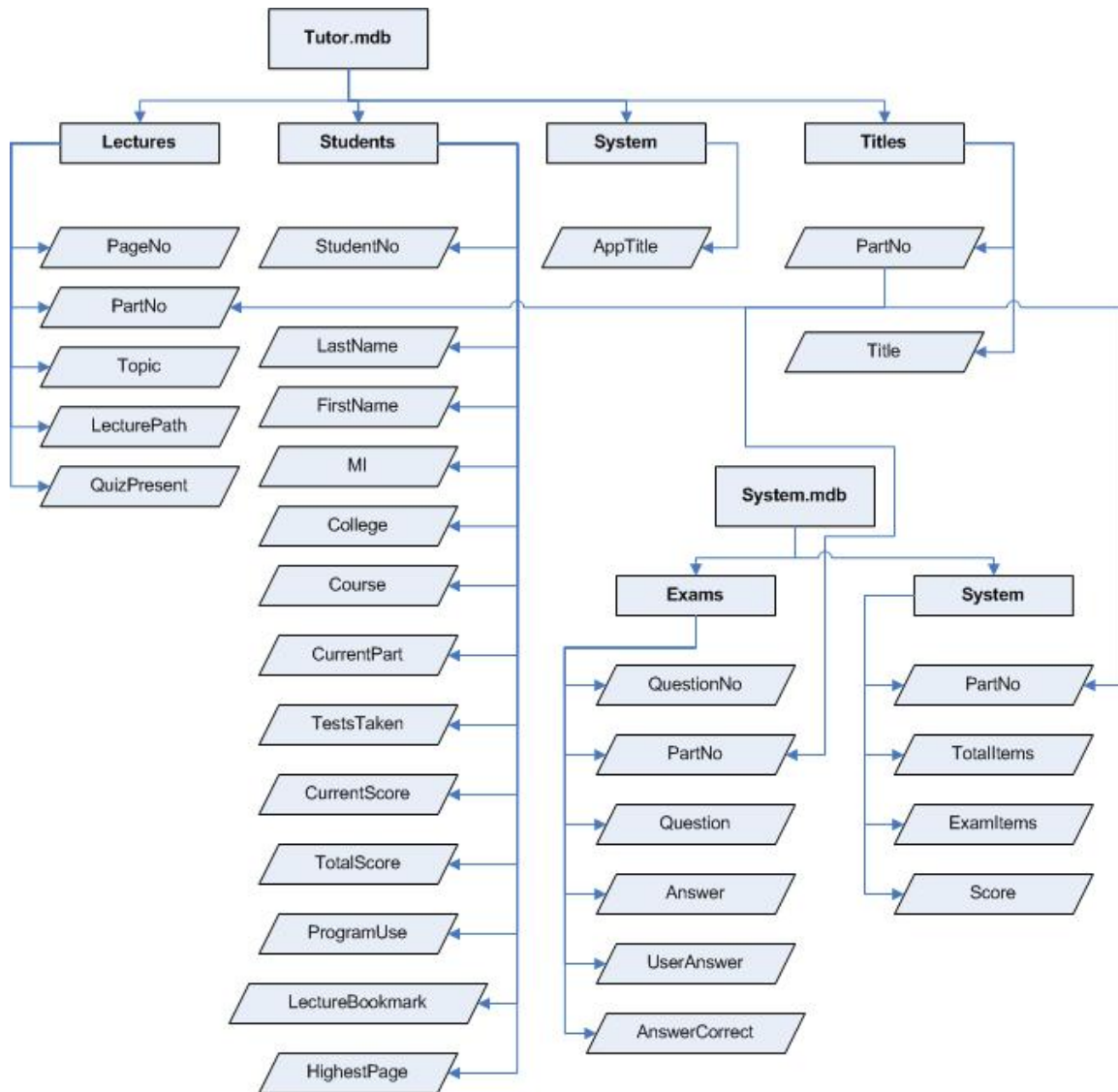


Figure 1: Database Structure of Tutor

Log in Screen (Fig. 2) – A log in system was implemented so that many students could share a single computer. The log in system permits every student to have their own profile so that their progress on the Tutor CAI could be exclusively set. A student who does not have a profile yet could click the Create New User button. Another window would be displayed that will ask some information about the user.

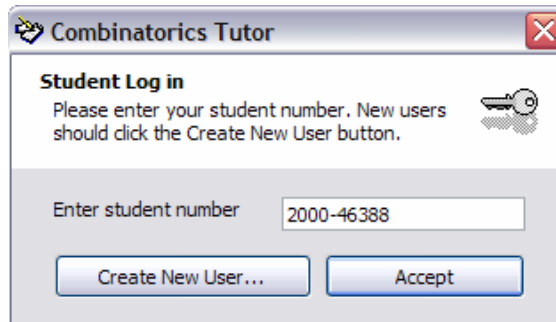


Figure 2: Combinatorics Tutor Log in Screen

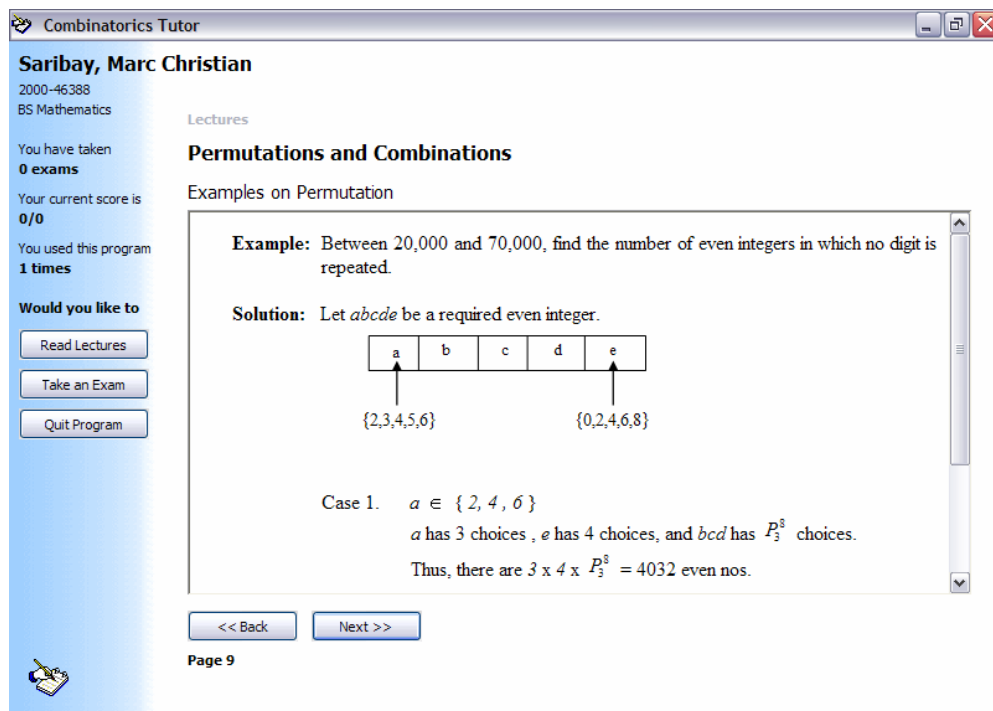


Figure 3: Combinatorics Tutor Screenshot

GUI (Graphical User Interface) Layout (Fig. 3) – The Tutor interface was designed in Visual Basic and was started in the form of a blank window. Then controls (buttons, text boxes, etc.) were added to the window in order to give a user-friendly interface. The GUI has been carefully thought of since it would present the usefulness of the program. A basic GUI which offers a fresh interface has been implemented so that even beginners could easily and quickly interact with the program. The GUI also supports the *Visual Styles* when it is installed in a Windows XP system through the use of a manifest file. Manifest files enhance a program by implementing the Visual Styles of Windows XP. A background picture was also

added. The interface was kept basic so that even users without experience with computers could easily maneuver his way around the CAI. The right pane shows the name of the student and some information on the student. It also shows the status of the student regarding the exams. The lower portion of the right pane presents the access buttons for lectures, exams, as well as the quit program button. Depending on the access button clicked by the user, the left pane changes in two ways. It could be a lecture pane or an exam pane.

Tutor (Front-end) – This is the core program of the Combinatorics Tutor designed to serve as a template for other fields of study. It was presented with empty databases, a blank HTML lecture, and a blank HTML exam. Included in the installation CD is a manual (Tutor Database Manual - in HTML format) to implement and use the Tutor (front-end) by simply manipulating the database (using Microsoft Access 97 or a later version) and designing the lecture and exam pages (using Microsoft Word 2000 or a later version).



Figure 4: Combinatorics Tutor Navigation Buttons Screenshot

Navigation System –The Back and Next buttons (Fig. 4) allows the user to have his own pacing while navigating through the lectures and exams. A user is allowed to backtrack on the lecture and exam pages. This was intended to provide learner-controlled CAI.

Lecture Bookmark and Non-trappable Quiz feature – Tutor was designed to remember the last lecture page that the student left. Also, to avoid a student from being trapped on a quiz (Fig. 5) and make him unable to proceed on further lecture pages, a feature was implemented which lets a student proceed to the next lecture page at next log-in even without entering a correct answer to the quiz.

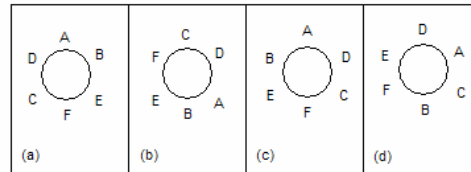
Lectures

Permutations and Combinations

Quiz

If six people, designated as A, B, C, ..., F, are seated about a round table, how many different circular arrangements are possible if arrangements are considered the same when one can be obtained from the other by rotation?

NOTE: Arrangements (a) and (b) are considered identical, whereas (b), (c), and (d) are three distinct arrangements.



<< Back

Next >>

Take the Quiz

Page 15

Figure 5: Combinatorics Tutor Quiz Screenshot

Creation of Lectures and Exams – The lectures and exams were created as HTML pages through the use of the Save As Web Page feature of Microsoft Office Word 2003 to enable support for mathematical symbols and advanced formatting. (Microsoft Word 2000 and later versions also supports the Save As Web Page feature). Tutor is not limited to the Microsoft Word Web Page Format. It can use any Web authoring programs for creating its Lecture and Exam pages in any Web Page format as long as the Internet Explorer browser supports it. Refer to the Tutor Database Manual for more details.

Exam Randomization – Tutor also supports random exam generation by utilizing the Randomize method in Visual Basic. The exam has a separate database so that duplicating it will conserve disk space. When a user takes an exam, a new database will be created and its contents will be picked from the exam database using the generated random numbers. The exam items on the exam database exceed the actual exam items for better randomization. The answers of the student will reflect on the generated exam database. When the exam is already submitted, the score of the student will be updated on the student database.

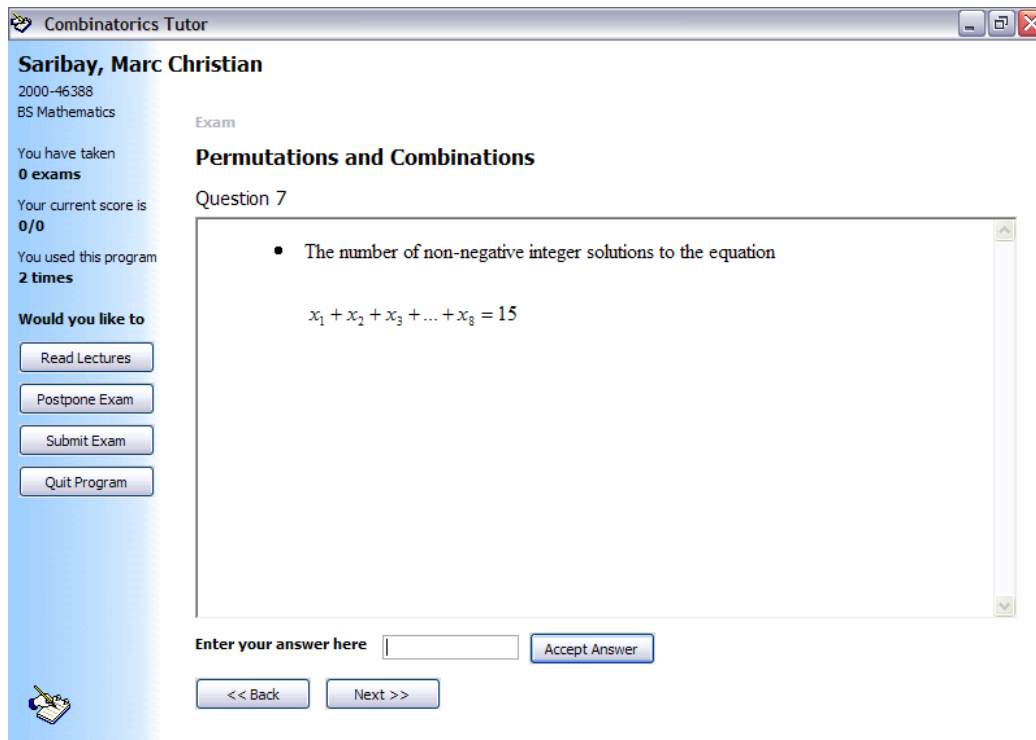


Figure 6: Combinatorics Tutor Exam Screenshot

Submit or Postpone Exam – The exam (Fig. 6) can be submitted or postponed anytime the student wishes. The student can leave some questions unanswered but no score will be credited on those questions when the exam has been submitted. Once the exam is submitted, the student's score will be automatically updated on the database. Also, the student is permitted to browse the lectures while having an exam.



About Dialog Window – Clicking the small icon in the lower left portion of the main window will show the About dialog window. This window was designed to contain some details regarding the project and some information about the author. A background sound (midi file) will be played upon opening this dialog.

Tutor Database Manual – This is the manual for the Tutor CAI authoring system which contains the instructions for the creation of CAI contents for Tutor. The Tutor Database Manual was printed in the following pages.

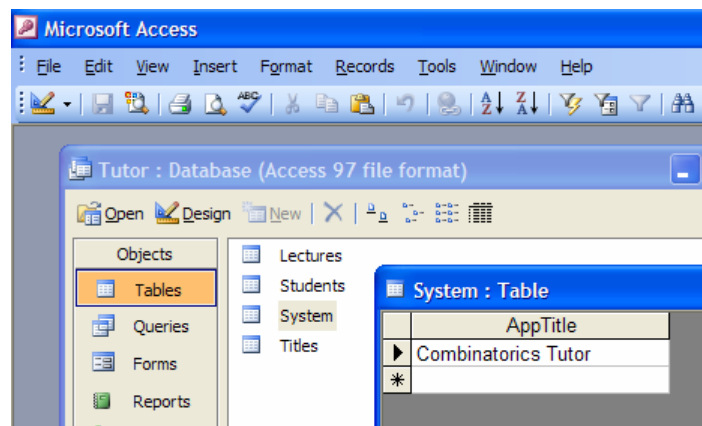
Tutor Database Manual

for Tutor 1.1.0

© 2004 by Marc Christian Saribay

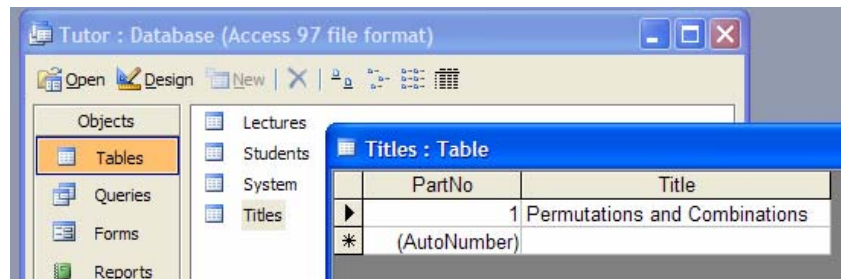
NOTE: *Microsoft Access 97* or a later version of *Microsoft Access* is required for database manipulation. *Microsoft Word 2000* or a later version of *Microsoft Word* is necessary for easy creation of HTML pages.

Customize the Program Title



- The title of the Tutor program can be changed to suit the field of study that will be lectured. Open the database **Tutor.mdb** on the path <Program Files folder>\Tutor\System using Microsoft Access. Open the **System** table. Type the custom title on the **AppTitle** field and make sure that it is the first and only record.

Organizing the Study into Parts



- Organize the whole study into several parts such that each part will have an exam. Now open the **Titles** table and enter the Title for every part in the **Title** field.

- Plan the number of exam items for each part.

Lecture Pages (Overview)

- Lecture pages will contain the discussion of the study. A lecture page has a **Topic**.
- A lecture page may or may not contain a **Quiz**. Quizzes are just self-assessment questions. If the student provided an incorrect answer, the student will be blocked temporarily from accessing the next lecture page.

Creating Lecture Pages

- Create the content in **Microsoft Word**. You can either choose to begin with a new **Web Page** or simply begin with a new Document and use the **Save As Web Page** feature.
- Writing equations are also supported. Microsoft Word includes **Equation Editor** but is not installed by default. You will need to install it first by running the setup program of Microsoft Word/Office.
- Every page of the content for the Tutor must not be long.
- A quiz can be integrated on a lecture page. Just state the quiz (preferably a whole page or on the lower part of the page) and make sure that the answer to the quiz is an integer. Take note of the answer with the corresponding lecture page. It will be handy later when integrating the lecture pages into the Tutor database.
- Save every page as a Web Page/HTML file.
- Use this save path for a lecture page:
<Program Files folder>\Tutor\Documents
- A good convention for naming every page for the Lecture would be Page1.htm, Page2.htm, ... and so on.

Integrating the Lecture Pages to the Tutor Database

- Now that the lecture contents are already formatted as Web Pages, it is time to integrate these pages into the Tutor database. To do this, open Tutor.mdb on the path <Program Files folder>\Tutor\System using Microsoft Access. Open the Lectures table.
-

PageNo	PartNo	Topic	LecturePath	QuizPresent	QuizAnswer
1	1	Addition Principle (AP)	Page1.htm	0	0
2	1	Examples on AP	Page2.htm	1	10
3	1	Multiplication Principle (MP)	Page3.mht	0	0
4	1	Examples on MP	Page4.htm	1	20
5	1	Permutations	Page5.mht	1	30
6	1	Examples on Permutations	Page6.mht	0	0
*(AutoNumber)	0			0	0

- Indicate the **PartNo** for the lecture page. The **PartNo** field corresponds to the parts of the study. For example, if the lecture page with the lecture topic Addition Principle belongs to Part 1 which is Combinations and Permutations (on the Titles table), enter 1 on the PartNo field.
- Enter the title of the topic in the **Topic** field for each lecture page.
- Enter the filename of the Web Page containing the lecture page on the **LecturePath** field. Just make sure that it is saved correctly on the path <Program Files folder>\Tutor\Documents.
- From time to time, a quiz can support the progress of the student. If a quiz is present in the lecture page, indicate it by entering 1 in the **QuizPresent** field.
- Enter the answer to the quiz on the **QuizAnswer** field.

Student Profiles

- The **Student** table contains all the student information and progress upon using the Tutor. There is no need to edit the entries on these fields unless necessary.

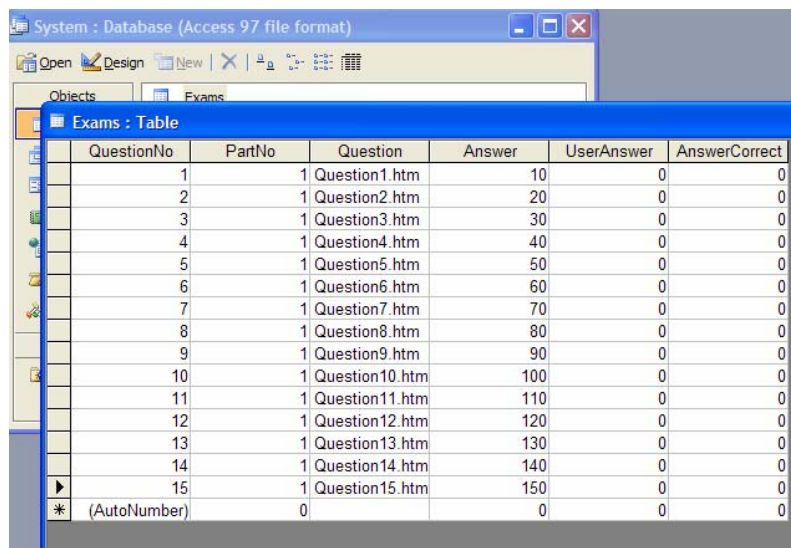
Exam Items (Overview)

- An exam item contains a single question.
- If the number of exam items has already been planned, create a set of exam items that will exceed the planned number of exam items. This will enable the randomization of exam items by just choosing a subset (planned exam items) from the set of available exam items.

Creating Exam Items

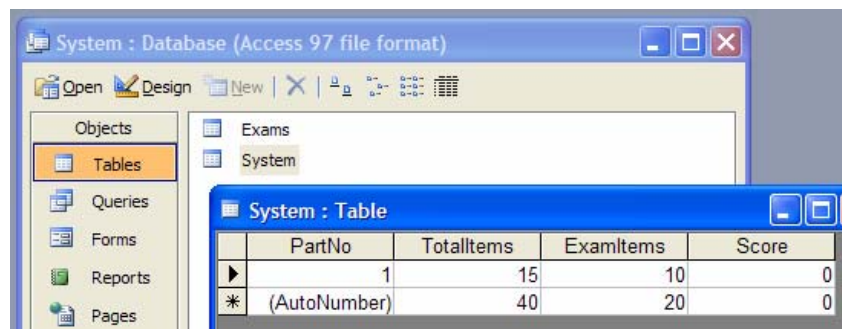
- Create the content in **Microsoft Word**. You can either choose to begin with a new **Web Page** or simply begin with a new **Document** and use the **Save As Web Page** feature.
- Writing equations are also supported. Microsoft Word includes **Equation Editor** but is not installed by default. You will need to install it first by running the setup program of Microsoft Word/Office.
- Every exam item for the Tutor must not be long. If the answer to the exam item is not an integer, then a multiple-choice type will be appropriate. State the choices and number them. Take note of the answer with the corresponding exam item. It will be handy later when integrating the exam items into the Tutor database.
- Use this save path for an exam item:
<Program Files Folder>\Tutor\Documents\System
- A good convention for naming every item for the Exam would be Question1.htm, Question2.htm, ... and so on.
- Create exam items which are more than needed. For example, if the plan is a 10 item exam, then create 15 exam items or higher. That way, the randomization of exam items feature of Tutor works better.

Integrating Exam Items into the Tutor Database



QuestionNo	PartNo	Question	Answer	UserAnswer	AnswerCorrect
1	1	Question1.htm	10	0	0
2	1	Question2.htm	20	0	0
3	1	Question3.htm	30	0	0
4	1	Question4.htm	40	0	0
5	1	Question5.htm	50	0	0
6	1	Question6.htm	60	0	0
7	1	Question7.htm	70	0	0
8	1	Question8.htm	80	0	0
9	1	Question9.htm	90	0	0
10	1	Question10.htm	100	0	0
11	1	Question11.htm	110	0	0
12	1	Question12.htm	120	0	0
13	1	Question13.htm	130	0	0
14	1	Question14.htm	140	0	0
15	1	Question15.htm	150	0	0
*	(AutoNumber)	0	0	0	0

- Open the database **System.mdb** on the path <Program Files folder>\Tutor\System using Microsoft Access. Open the **Exams** table. Indicate the **PartNo** of the exam item. The **PartNo** field corresponds to the parts of the study. For example, if the exam item Question 1 belongs to Part 1 which is Combinations and Permutations (on the **Titles** table), enter 1 on the **PartNo** field.
- Enter the filename of the Web Page containing the exam item on the **Question** field. Just make sure that it is saved correctly on the path <Program Files folder>\Tutor\Documents\System.
- Enter the answer to the exam item on the **Answer** field.



- Now open the database **System.mdb** on the path <Program Files folder>\Tutor\System using Microsoft Access. Open the **System** table. Enter the total number of exam items that is created for the corresponding part (indicated by the **PartNo** field) on the **TotalItems** field.
- Indicate the actual number of exam items on the **ExamItems** field. The actual number of exam items should be less than the total of created exam items.

Understanding the Exam Mechanism

- A new database file will be generated when a student takes an exam. The filename is in the format <Student Number>exam<exam part>.mdb and will reside on the path <Program Files folder>\Tutor\System.
- The generated database will contain random exam items and will be updated with the student's answers to the exam on the **Exams** table. The **System** table will also be updated with the student's score upon submitting the exam.

Combinatorics Kit

Combinatorics Kit 2 – This is the second release of the Combinatorics Kit that was included to supplement the Combinatorics Tutor CAI. The feature set of the Combinatorics Kit 2 program consists of the following combinatorial methods: *Factorial, Permutation, Combination (with list generation), Circular Permutation, Stirling Number of the 1st Kind (with list generation), Stirling Number of the 2nd Kind (with list generation), r-element multi-subset of n, Pairing, Arrangement with repetitions, Selection of objects having some objects of the same type, k-ary sequences of length n, Positive divisors of a number, Distribution methods, Bell Number, Derangements, and Lexicographic Ordering.*

System Design of Combinatorics Kit 2 (Fig. 7) – The Combinatorics Kit 2 project file in Visual Basic 6 contains two main files. frmMain.frm is a Form file which includes some local functions and subroutines as well as the interface data of the Combinatorics Kit window. modMain.bas is a Module file which includes global functions, subroutines, and variable declarations. A Module file does not contain an interface data. The system design flowchart of Combinatorics Kit 2 is presented on the following page. All of the combinatorial methods which were already transformed into algorithms in the form of functions or subroutines are placed in the Module file. The Form file which includes the interface data would be watching the events of the controls at runtime. When an event is triggered, which is usually a click event, the control which triggered the event would call the needed functions or subroutines from the Module file. For example, if a control which is named cmdMethod(10) corresponds to a command button with the caption “Pairing” on the Form interface is clicked, it will call the Function Pair (Pairing method) from the Module file using the parameter n which is an input received from the Form interface. The Function Pair will then need to call Function Fact (Factorial Method) to proceed with the necessary output. Any Form files in the current project could access the module files of the project because module files are globally accessible.

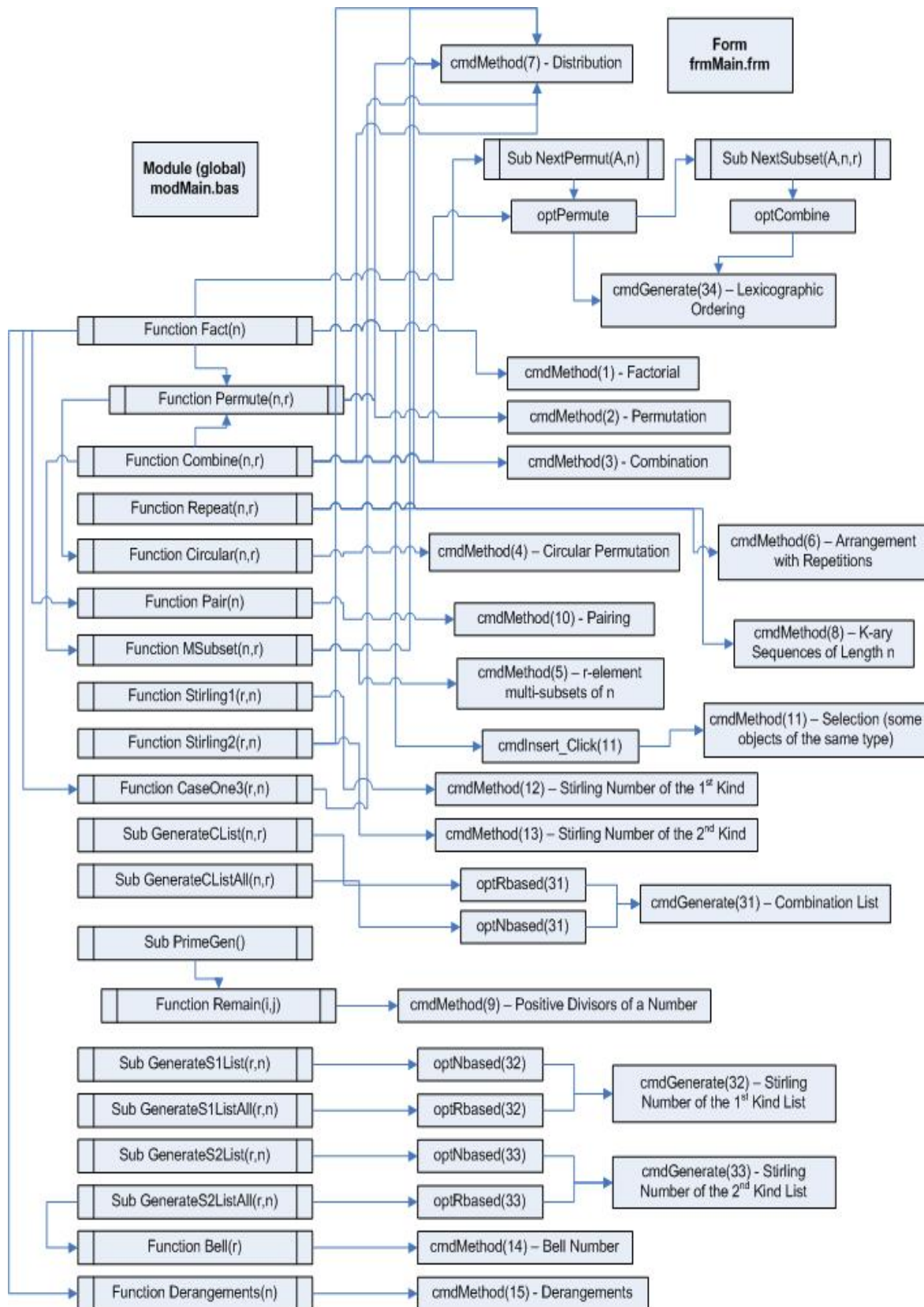


Figure 7: System Design of Combinatorics Kit 2

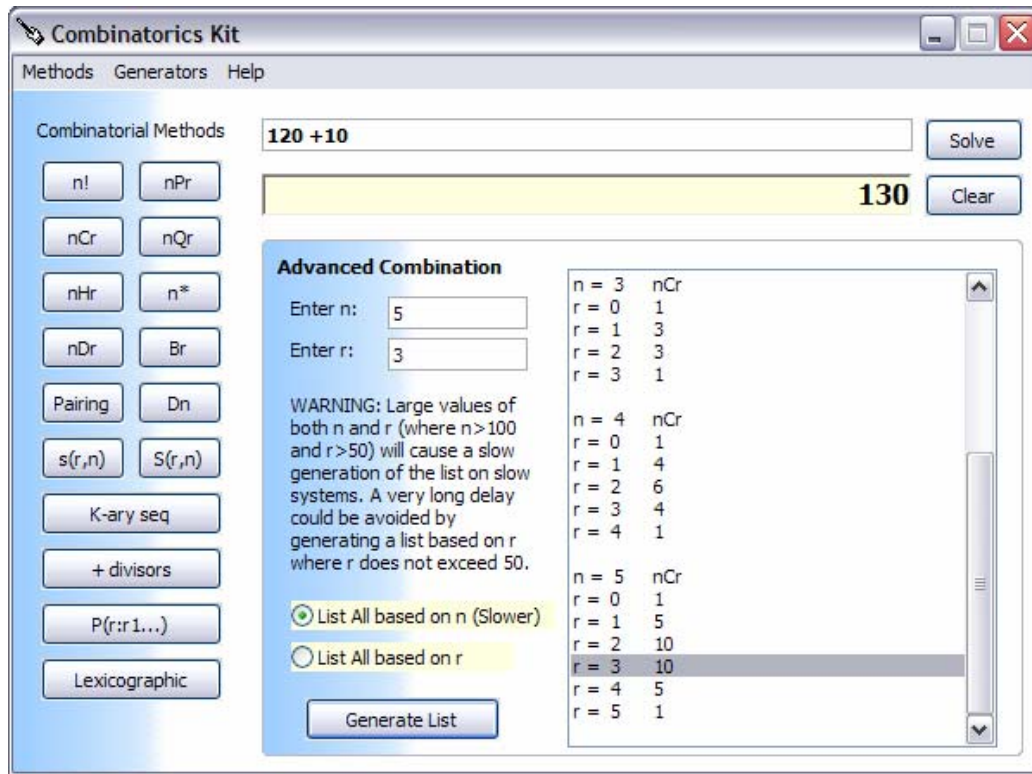


Figure 8: Combinatorics Kit 2 Screenshot

GUI (Graphical User Interface) Layout (Fig. 8) – The Combinatorics Kit Visual Basic project was started by creating its interface in the form of a blank window. Then controls (buttons, text boxes, etc.) were added to the window in order to give a user-friendly interface. The GUI has been carefully thought of since it would present the usefulness of the program. The GUI has been redesigned so that the user would be able to interact with the program quickly, easily, and efficiently without the window cluttering which is found on the previous version. The program now presents a fresh and streamlined interface. The GUI also supports the *Visual Styles* when it is installed in a Windows XP system through the use of a manifest file. Manifest files enhance a program by implementing the Visual Styles of Windows XP. Some buttons corresponding to each combinatorial method were placed in the left pane of the window. The Solver was placed on the upper pane. The right pane, which is located below the Solver, will display a form which will get the necessary inputs depending on the chosen combinatorial method and will change accordingly. A background picture was also added. At first, it was handled

at design time and resulted into an increase in the size of the EXE file because every combinatorial method frame also displays the background picture. It was modified so that the graphic file will just be loaded at runtime. The end result was an efficient program which loads the graphic file only when necessary.

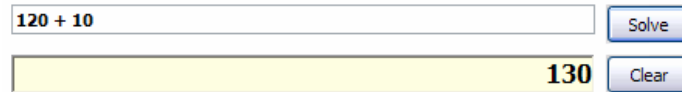


Figure 9: The Solver

The Solver (Fig. 9) – The Solver occupies the upper portion of the window. It was designed to function as a built-in syntax-based calculator. It was implemented using the Script control in Visual Basic. The Solver would accept the result of every combinatorial method used. The syntax could be edited to speed up computations involving operations of various combinatorial methods. Clicking the Solve button would analyze the syntax and display the result.

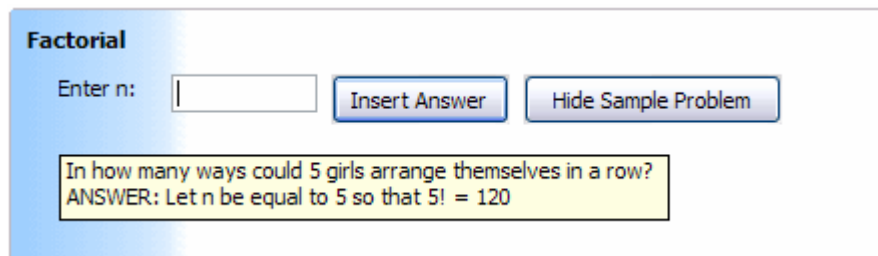




Figure 10: Factorial Method Screenshot

 Factorial Method (Fig. 10) – A Factorial method window would require a value for n . Since the Double data type of variables in Visual Basic fails to handle values exceeding $1.79769313486232 \times 10^{308}$, the factorial method could only accept up to $170!$. Note that an ordinary scientific calculator already starts to fail at $70!$

 Permutation Method (Fig. 11) – A Permutation method window would require values for n and r . The Permutation method depends on the factorial of r (r

up to 170 only). Since this method was based on the Combination method subroutine, n could take large values.

Permutation

Enter n:

Enter r:

In how many ways could 5 boys arrange themselves in a row with 3 seats only?
ANSWER: Let n be equal to 5 and let r be equal to 3 so that $P(5,3) = 60$

Figure 11: Permutation Method Screenshot

Circular Permutation

Enter n:

Enter r:

In how many ways can 4 persons sit on a round table consisting only 3 seats?
ANSWER: Let n be equal to 4 and let r be equal to 3 so that $Q(4,3) = 8$

Figure 12: Circular Permutation Method Screenshot

Circular Permutation Method (Fig. 12) – A Circular Permutation method window would require values for n and r . This method was based on the Permutation method. Therefore, same limitation on r factorial was expected.

Arrangement with Repetitions and K-ary Sequences of Length n Methods (Fig. 13) – A single set of code was shared between these two methods. The set of restrictions were different though.

Arrangement with Repetitions

Enter n:

Enter r:

How many combinations are there in a 3-digit lock if a digit ranges from 0 to 9?
ANSWER: Let n be equal to 10 and let r be equal to 3 so that $10^3 = 1000$

Figure 13: Arrangement with Repetitions Method Screenshot

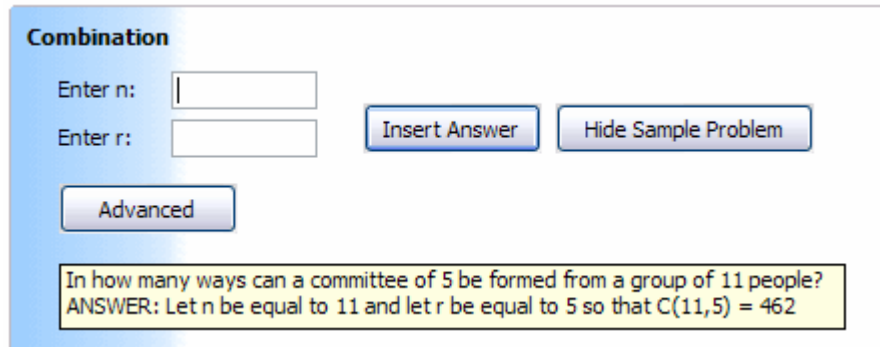


Figure 14: Combination Method Screenshot

nCr Combination Method (Fig. 14) – The previous algorithm which relied on the generation of the Pascal's triangle was slow. When the algorithm was revised, faster performance was noticed. A part of the Visual Basic revised code for the Combination method was placed below. This method would be flexible enough to handle large values and still return the result quickly.

```

Dim I, j, Temp As Double
Temp = 1
If r = 0 Or r = n Then
    Combine = 1
ElseIf r = 1 Or r = n - 1 Then
    Combine = n
Else
    If r <= (n - r) Then
        For I = 1 To r
            Temp = Temp * ((n - r + i) / i)
        Next
    ElseIf r > (n - r) Then
        For I = 1 To (n - r)
            Temp = Temp * ((n - (n - r) + i) / i)
        Next
    End If
    Combine = Temp
End If

```

[+ divisors](#) Positive Divisors of a Number Method (Fig. 15) – Two major steps were implemented in this method. The first step was the generation of the prime numbers. This method has been limited to the 168th prime number (still large enough). The second step was the prime factorization algorithm.

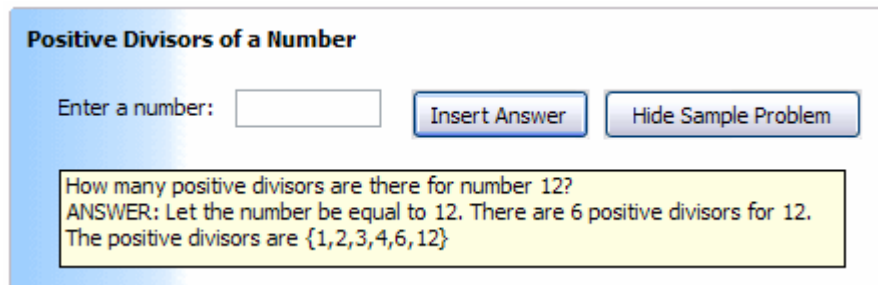


Figure 15: Positive Divisors of a Number Method Screenshot

[Pairing](#) Pairing Method (Fig. 16) – This method could only accept up to 85 pairs due to the $(2n)!$ limitation.

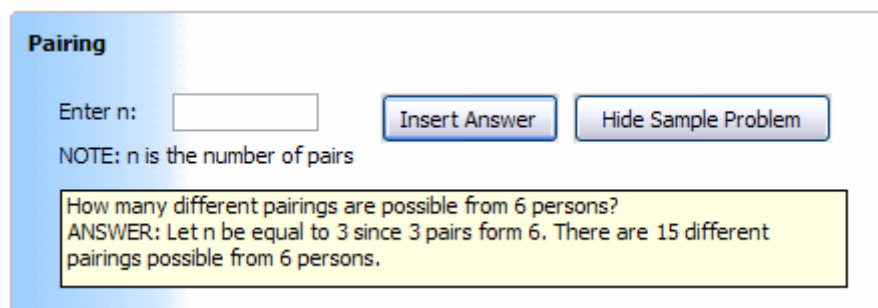


Figure 16: Pairing Method Screenshot

About Dialog Window – This window was designed to contain some details regarding the project and some information about the author. A background sound (midi file) will be played upon opening this dialog.

[Show Sample Problem](#) Sample Problems – Some combinatorial methods have been modified to contain a new button called “Sample Problem.” This button will be a big help for those users having problems on the application of a certain combinatorial method.

Combination List/Table Generators (Fig. 17) – The Combination Table generator part of the project was scrapped due to the overlapping columns problem on very large values. Although the “table” gives a better format for smaller values, the “list” format presented every value (small or large) nicely. Updates presented in the generators include the “Please Wait...” indication and mouse pointer that changes into an hourglass icon while generating the list. The screen corruption problem upon generation of the list which is present on the previous version was now fixed.

Advanced Combination

Enter n:

Enter r:

WARNING: Large values of both n and r (where $n > 100$ and $r > 50$) will cause a slow generation of the list on slow systems. A very long delay could be avoided by generating a list based on r where r does not exceed 50.

☒ List All based on n (Slower)

☐ List All based on r

n	nCr
n = 3	
r = 0	1
r = 1	3
r = 2	3
r = 3	1
n = 4	
r = 0	1
r = 1	4
r = 2	6
r = 3	4
r = 4	1
n = 5	
r = 0	1
r = 1	5
r = 2	10
r = 3	10
r = 4	5
r = 5	1

Figure 17: Combination Generator Screenshot

nHr r-element Multisubset of n Method (Fig. 18) – Just like the other methods, the result of this method was limited by the capacity of the Double data type variable. This method was free from the $n \geq r$ restriction.

(nHr) r-element Multi-subsets of n

Enter n:

Enter r:

Assuming that there are no limits on the supply of each type of sandwich, how many orders can the boy place if he is going to order 6 sandwiches from 3 types of sandwiches?

ANSWER: Let n be equal to 3 and let r be equal to 6 so that $H(3,6) = 28$

Figure 18: r-element Multisubset of n Method Screenshot

$s(r,n)$ $S(r,n)$ Stirling Numbers (1st and 2nd kind) Methods (Fig. 19 and Fig. 20 respectively) – The recursive formula was applied in this method. When this method is called, the values are being generated from the beginning until the desired result is reached. It was a slightly slow approach but there is no other way.

Stirling Number of the 1st Kind

Enter r:

Enter n:

If there must be at least one person in each table, in how many ways can 6 people be seated around three tables?
ANSWER: Let r be equal to 6 and let n be equal to 3 so that $s(6,3) = 225$

Figure 19: Stirling Number of the 1st Kind Method Screenshot

Stirling Number of the 2nd Kind

Enter r:

Enter n:

In how many ways can 5 different objects be placed into 2 identical boxes such that no box is empty?
ANSWER: Let r be equal to 5 and let n be equal to 2 so that $S(5,2) = 15$

Figure 20: Stirling Number of the 2nd Kind Method Screenshot

Restriction Notation (Fig. 21) – The restriction notation was revised. The previous notation made use of words (“n greater than or equal to r”). It was now converted to symbol notation (“ $n \geq r$ ”).



Figure 21: Restriction Notation Screenshot

Stirling Number (1st and 2nd kind) List/Table Generators (Fig. 22 and Fig. 23 respectively) – The recursive formula was perfectly applicable for this method. Plans for “table” generators were discontinued because of the overlapping columns problem and instead, focused on the “list.” The Tables menu was also removed. Updates presented in the generators include the “Please Wait...” indication and mouse pointer that changes into an hourglass icon while generating the list. The screen corruption problem upon generation of the list which is present on the previous version was now fixed. Another update was the error indication. It was modified so that when an error for "large values encountered" occurs upon list generation, the list is never cleared. Instead, the error was modified to state that the desired values were not met. A formula bug was found in the Stirling Number of the 2nd Kind list generation based on n and was fixed. This major bug is present in the previous version and displays erroneous results on the list.

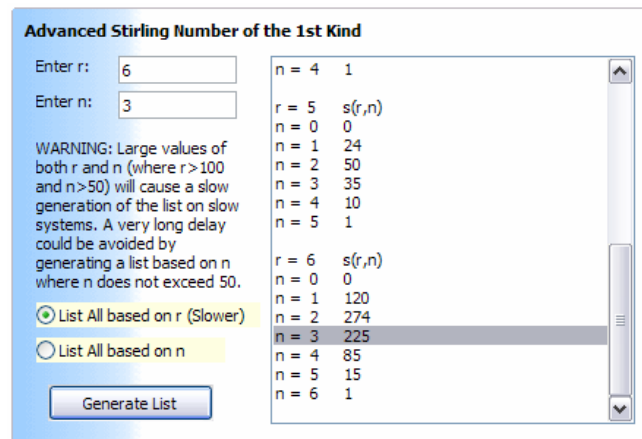


Figure 22: Stirling Number of the 1st Kind Generator Screenshot

Advanced Stirling Number of the 2nd Kind

Enter r:

Enter n:

WARNING: Large values of both r and n (where $r > 100$ and $n > 50$) will cause a slow generation of the list on slow systems. A very long delay could be avoided by generating a list based on n where n does not exceed 50.

☒ List All based on r (Slower)

☐ List All based on n

n = 1	1
n = 2	3
n = 3	1
r = 4 $S(r,n)$	
n = 0	0
n = 1	1
n = 2	7
n = 3	6
n = 4	1
r = 5 $S(r,n)$	
n = 0	0
n = 1	1
n = 2	15
n = 3	25
n = 4	10
n = 5	1

Figure 23: Stirling Number of the 2nd Kind Generator Screenshot

Br Bell Number (Fig. 24) – The first code computes for every Stirling Number of the second kind which is generated by a loop and then its summation is performed. This algorithm would tend to get noticeably slow if large values are given. Thus, the code was revised to utilize the Stirling Number of the 2nd Kind list generation based on r and was modified to just locate the necessary row and take its sum. It was faster and more efficient than the previous code.

Bell Number

Enter r:

What is the number of partitions of a set with 4 elements?
ANSWER: Let r be equal to 4 so that $B(4)=15$

Figure 24: Bell Number Method Screenshot

Dn Derangements (Fig. 25) – This method is a formula which required a loop and was limited by the factorial method. Reaching large values would never be a problem due to the factorial restriction.

Derangements

Enter n:

What is the number of derangements of the set with 5 elements?
ANSWER: Let n be equal to 5 so that $D(5)=44$

Figure 25: Derangements Method Screenshot

Lexicographic

Lexicographic Ordering (Fig. 26) – Two options were introduced in this generator. The first one enumerates all the possible permutations of n elements. The second option enumerates all the possible combinations of r elements which are taken from n elements. Generating the permutations of 8 elements already causes a long delay on a 1.5 Ghz Pentium 4 system. A warning message will be present to avoid such delays. It was restricted to 100 elements only but was not restricted on the enumerated output (the user may choose to continue even if the delay was very long) to avoid defeating the purpose of this generator.

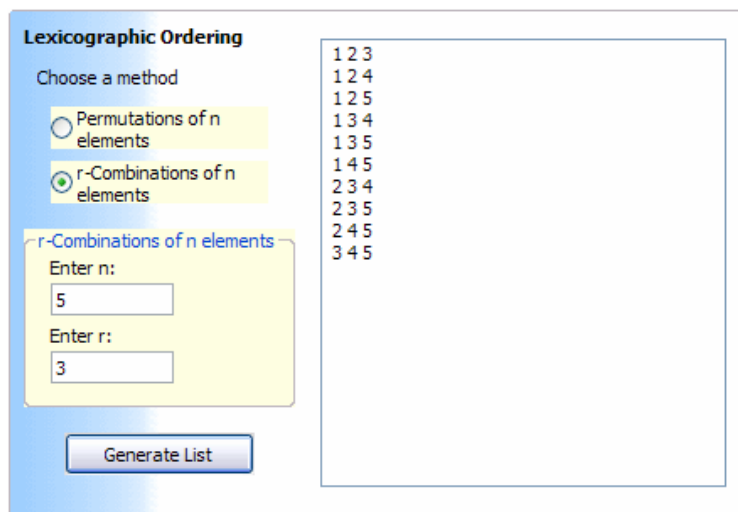


Figure 26: Lexicographic Ordering Screenshot

P(r:r1...)

Selections with Objects of the Same Type Method (Fig. 27) – This method was limited to handle up to 20 different types/classifications only. This method went through two revisions. The second revision of this method required the numerator value. This way, the sum of the denominators could be checked so that it would never exceed the numerator. Updates presented in this method are some minor bug fixes. A bug which could bypass the numerator requirement is fixed. Another bug which allows assignment of a letter to the numerator and gives a value too large error was also fixed. These bugs exist in the previous version.

Figure 27: Selections with Objects of the Same Type Method Screenshot

nDr

Distribution Method (Fig. 28) – This method was a compilation of seven methods. It was designed to offer quick results by using the pre-defined conditions and choosing the proper option for the similarity of the boxes and the objects.

Figure 28: Distribution Method

Copy Answer Trick – Two methods were created to copy the current result of the Solver. The quick way was to click on the result. The other way was to click the Copy Answer from the Method menu.

Generation Delay Warning Dialog (Fig. 29) – This warning message was implemented in the program so that the user will be able to avoid very long delays (especially on slow computers) upon using the list generators.

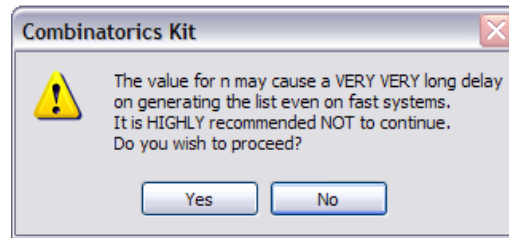


Figure 29: Generation Delay Warning Dialog Screenshot

“Please Wait...” Message and Hourglass Mouse Pointer – On all generators, the Generate List button changes its caption to “Please wait...” and the mouse pointer changes into an hourglass icon while generating a list. This feature is implemented in the program to let the user know that the list is being generated. This feature is not present on the previous version.

Help System (Fig. 30) – The HTML Help system serves as a guide on using the program. (F1 key activates the Help System; Internet Explorer 4.0 or later required). The help system was revised to contain the new features of the second version.

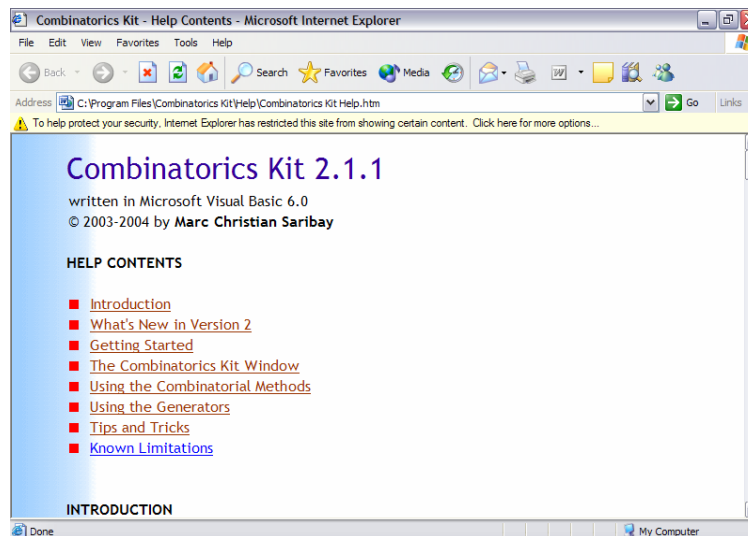


Figure 30: Help System Screenshot

Combinatorics Kit Plus CD

Combinatorics Kit Installation – After loading the Combinatorics Kit Plus CD on the disc tray of the CD drive, it would automatically run the setup program for the Combinatorics Kit program installation. (The Autoplay setting must be enabled within Windows)

Combinatorics Tutor Installation – Browse to the Combinatorics Tutor Setup folder on the installation CD. Run Setup inside this folder.

Tutor Installation – Browse to the Tutor Setup folder inside the Bonus Software folder on the installation CD. Run Setup in the Tutor Setup folder. It will install the Tutor front-end that contains no lectures and exam. This is intended for creating a computer-aided learning interface on other fields of study. A manual for the Tutor front-end (Tutor Database Manual – in HTML format) is also included on the Tutor Setup folder of the installation CD. This manual helps on providing the necessary settings to make the Tutor work on the chosen field of study. Database manipulation requires Microsoft Access 97 or a later version. Creation of Lecture and Exam pages requires Microsoft Word 2000 or a later version.

Readme Files – These text files contain the version history and installation instructions of the program.

CHAPTER VI: SUMMARY AND CONCLUSION

Combinatorics Tutor introduces a computer-aided learning interface for students and handles lectures, quizzes and exams on Combinatorics for each user. It is also designed to generate random exams. Its exam database (main set) contains more exam items than the actual exam items (getting a subset of the main set) for better randomization.

Tutor (Front-end) is the core program of the Combinatorics Tutor and is designed to serve as an authoring template for other fields of study. It was presented with empty databases, a blank HTML lecture, and a blank HTML exam.

Exam answers and Quiz answers in Tutor (Front-end) currently support only numeric values. A multiple choice type of questions with numbered choices will be a good workaround for questions that does not need a numeric answer.

The Combinatorics Kit GUI has been redesigned so that the user would be able to interact with the program quickly, easily, and efficiently without the window cluttering which is found on the previous version. The program now presents a fresh and streamlined interface.

Factorial method was restricted from exceeding $170!$ due to the limitations of double-precision variable data type. Permutation and Circular Permutation methods were based on combination and the limitation of r factorial. Combination method was based on a good algorithm which could handle parameters having large values. But the process is being limited by the variables due to the double-precision data type. (except for the properties $nC_0=1$ and $nC_n=1$)

Pairing method was limited up to $n=85$ pairs. Number of Positive Divisors method was limited up to the 168th prime number (generating more prime

numbers which exceeds the 168th prime number limit will make the loading of the window slower).

Stirling numbers of the 1st and 2nd kind as well as the “n to the r” methods used by Arrangement with Repetitions and K-ary Sequences of Length n is being limited by the highest double-precision value that the variables of the double-precision data type can handle

The Selections with Objects of the Same Type method has been limited to 20 types/classifications only. Bell Number method used the Stirling Number of the 2nd Kind list generation based on r method and took the sum of the required row for fast execution. Derangements method was limited by the factorial restriction.

Lexicographic Ordering has been restricted to 100 elements on both options but was not restricted on the enumerated output (the user may choose to continue even if the delay was very long) to avoid defeating the purpose of this generator. Generating a list with more than 5000 possibilities could be done but would need a great amount of time.

Some combinatorial methods have been modified to contain a new button called “Sample Problem.” This button will be a big help for those users having problems on the application of a certain combinatorial method.

The limitations listed here are due to the limitations of the variables declared to be of double-precision data type. Even so, this program is far more capable since it could accept higher values which are far beyond the capacity of current scientific calculators.

CHAPTER VII: RECOMMENDATIONS

Insufficient amount of time did not permit the author to perform the deployment of the Combinatorics Kit Plus for use. Field testing is recommended so that the effectivity and efficiency of the software could be evaluated. Also, HTML/Web pages could offer more interactive features such as animation, sound, and graphics which will make lectures more entertaining. Utilizing these features for the improvement of lessons is highly encouraged.

Regarding the randomization of exams, only 25 exam pages were created. The actual exam items consist only of 15 items. The randomizing feature of exams will work better upon creation of more exam pages. Thus, generating more exam pages will minimize the chance of repeating an exam item on other users.

Currently, the exam items could only accept numeric answers since the target field of study is Combinatorics. Multiple choice type of exam could be implemented provided that the choices are numbered. It would be better if answers in any format could be allowed.

As stated earlier, this study has been limited to include only the Permutations and Combinations part from the whole Combinatorics study. Since Tutor was designed as a CAI authoring system, inclusion of other topics to complete the whole Combinatorics study is greatly encouraged. Moreover, using Tutor in other fields of study is also encouraged.

CHAPTER VIII: REFERENCES

- Anderson, R. C. *Some reflections on the acquisition of knowledge*. Educational Researcher, 13, 5-10, 1984.
- Ausubel, D. P. *Educational psychology: A cognitive approach*. New York: Holt Rinehart and Winston, 1968.
- Berge, C. *Principles of Combinatorics*, Academic Press, New York, 1971.
- Brown, S. *Visual Basic 6 In Record Time*, SYBEX Inc., Alameda, CA, 1998.
- Brown, S. and E. Petroustos, *Visual Basic 6 Complete*, SYBEX Inc., Alameda, CA, 1999.
- Brualdi, R. *Introductory Combinatorics*, 2nd ed., North Holland, New York, 1991.
- "Computer-Aided Instruction (CAI)" *Microsoft ® Encarta ® Reference Library 2004*. © 1993-2003 Microsoft Corporation
- DeCharms, R. *Personal causation*. New York: Academic Press, 1968.
- Fein, G. G. *Pretend play in childhood: An integrative review*. *Child Development*, 52, 1095-1118, 1981.
- Grimaldi, *Discrete and Combinatorial Mathematics: An Applied Introduction*, 2nd ed.
- Hurlock, R. E., Lahey, G. F., McCann, P. H. *Student controlled versus program controlled CAI*. Paper presented at the meeting of the American Educational Research Association, Chicago. (ERIC Document Reproduction Service No. ED 089 681), 1974.
- Keller, J., Suzuki, K. *Use of the ARCS motivation model in courseware design*. In D. Johassen (Ed.), *Instructional designs for microcomputer courseware*, (pp. 401-434). Hillsdale, NJ: Earlbaum, 1988.
- Kinzie, M. *Requirements and benefits of effective interactive instruction: Learner control, self-regulation, and continuing motivation*. *Educational Technology Research and Development*, 38(1), 5-21, 1990.
- Kinzie, M. B., Sullivan, H. J., Berdel, R. L. *Learner control and achievement in science computer-assisted-instruction*. *Journal of Educational Psychology*, 80, 299-303, 1988.
- Lepper, M. R. *Microcomputers in education: Motivational and social issues*. *American Psychologist*, 40, 1-18, 1985.
- Lepper, M. R., Malone. *Intrinsic motivation and instructional effectiveness in computer-based education*. In R. E. Snow, M. J. Farr (Eds.) *Aptitude, learning, and instruction: III. Cognitive and affective process analyses* (pp. 255-296). Hillsdale, NJ: Erlbaum, 1987.
-

- Malone, T. *Toward a theory of intrinsically motivating instruction*. Cognitive Science, 4, 333-369, 1981.
- Mayer, R. E. *Learner-controlled instruction--1958-1964*. Programmed Instruction, 4(2), 1, 8, 10-12, 1964.
- Mayer, R. E. *Information processing variables in learning to solve problems*. Review of Educational Research, 4, 525-541, 1975.
- Mayer, R. E. *Aids to text comprehension*. Educational Psychologist, 19, 30-42, 1984.
- Muth, K. D. *Solving arithmetic word problems: Role of reading and computational skills*. Journal of Educational Psychology, 74, 199-216, 1984.
- Nijenhuis A. and H. Wilf, *Combinatorial Analysis*, John Wiley & Sons, New York, 1958.
- Padma, A. G., Ross, S. M. *Using computer-assisted-instruction to personalize arithmetic materials for elementary school children*. Journal of Educational Psychology, 79, 72-78, 1987.
- Parker, L. E., Lepper, M. R. *Effects of fantasy contexts on children's learning and motivation: making learning fun*. Journal of Personality, and Social Psychology, 62, 625-633, 1992.
- Reiber, L. P. *Animation, incidental learning and continuing motivation*. Journal of Educational Psychology, 83, 318-328, 1991.
- Roberts, F. *Applied Combinatorics*, Prentice-Hall, Eaglewood Cliffs, NJ, 1984.
- Rumelhart, D. E., Ortony, A. *The representation of knowledge in memory*. In R.C. Anderson, S. J. Sprio, W.E. Montague (Eds.), *Schooling and the acquisition of knowledge* (pp. 99-135), Hillsdale, NJ: Earlbaum, 1977.
- Singer, J. L. *Imagination and make-believe play in early childhood: Some educational implications*. Journal of Mental Imagery, 1, 127-144, 1977.
- Tennyson, R. D. *Instructional control strategies and content structure as design variables in concept acquisition using computer-based instruction*. Journal of Educational Psychology, 72, 525-532, 1980.
- Tucker, A. *Applied Combinatorics*, 3rd ed., John Wiley & Sons, New York, 1995.
-