To the Graduate Council:

I am submitting herewith a thesis written by A. Russell Jones III entitled "A Microcomputer Program to Facilitate Chord Spelling and Recognition." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Music.

Donald Pederson

Donald Pederson, Major Professor

We have read this thesis
and recommend its acceptance:

Stephen E Young

John A. Lam

Allen E Johnson

Accepted for the Council:

Carl Minkel

Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature A. Russell Jones III

Date 5-12-85

A MICROCOMPUTER PROGRAM TO FACILITATE CHORD SPELLING AND RECOGNITION

A Thesis

Presented for the

Master of Music

Degree

The University of Tennessee, Knoxville

A. Russell Jones III

August 1985

ACKNOWLEDGMENTS

I would like to express my appreciation to Professor Donald Pederson for his guidance and assistance during the course of my studies at The University of Tennessee.

I would also like to express my gratitude to my parents, Russell and Wanda Jones, for their support and encouragement.

I hereby acknowledge with thanks the use of the Higher Text MCI© utility program developed by Ron and Darrell Aldritch. (1980)

ABSTRACT

Most of the microcomputer software for music educators has been aimed at drill and testing of aural skills. The advancing capabilities of microcomputer graphics have opened the way for programs designed to teach visual music skills as well. The purpose of this study was to build a set of programs which could be used as support material for basic courses in music theory. Most previous music software has limited the material by allowing only lessons designed by the programmer to be used. This study consists of a set of three programs, one of which is a utility program designed so any music teacher can enter lessons specifically designed for his/her needs.

The three programs are designed to run on Apple IIe or II+ microcomputers with 64K of RAM memory. The instructor can use the software editor utility to control course content and the students use the other two programs, one designed to test (or drill) chord recognition, the other to test (or drill) chord spelling.

The programs allow students to proceed at their own pace. Records of progress in the form of test scores are kept for each student. All common-practice chords are supported by the programs except those with double-sharps or double-flats.

The chords appear in whole-notes on a grand staff. Both open and close position chords are accepted by the program. Since the chords are written in black notation on a white background, a color monitor is not necessary.

TABLE OF CONTENTS

A GLOSSARY OF USEFUL TERMS

Boot--To start the process of loading and running a program. On the Apple microcomputers, this process begins automatically when the machine is turned on. The disk operating system is loaded, and the HELLO program is run.

Computer-Assisted Instruction--A teaching method where a computer is used to check student answers and give corrections.

CRT monitor--A cathode ray tube contained in a box like a television set. The resolution is higher than that of a T.V. in order to achieve clearer text,

Disk--A disk, much like a phonograph record, which is covered with a magnetic substance on which programs and data are recorded.

DOS--Disk Operating System

FID--A program contained on the Applesoft DOS System Master Disk which is useful for transferring files from one disk to another.

Hard Disk--Another method of saving programs and data. These systems hold much more information than floppy disks, and retrieval time is usually much faster.

Hardware--Equipment for computers, including computers themselves.

HELLO--The program which is run automatically when the disk is booted.

IC--Integrated Circuit

Input--Data acquired by a program during the time it is running.

Menu--A set of choices presented to a user of a program.

Menu-driven input--Input which is acquired by a program through the use of menus.

Microcomputer--A self-contained, portable unit containing a microprocessor, support IC chips, and a typewriter-like keyboard.

Microprocessor--A small, but still powerful processor chip.

Processor--An IC chip which processes program instructions and data.

Program--A set of computer instructions.

RAM--Random-acess memory; memory which is located in the computer on IC memory chips. This memory can be altered by programs or user input.

ROM--Read-only memory; memory which can not be overwritten or changed.

Software--Programs available on disks, ROM or tape.

Utility--A program which is used to edit data files, or other programs.

INTRODUCTION

1. COMPUTER-ASSISTED INSTRUCTION IN MUSIC THEORY

The use of computers for theory-related courses is not new. Music educators have been using computers since the late 1960's. All of the early work was done with programs designed to run on large computers, but since the microcomputer has become an inexpensive, powerful tool, the main thrust in computer-assisted instruction (CAI) has been toward microcomputer based courseware. Microcomputers are affordable, portable, easy to maintain, and relatively easy to use. Unfortunately, music CAI software for microcomputers is still rare. Many programs that are commercially available are not suited for use by advanced students, or contain problems which are tailored to the needs of one specific school.

The idea of writing a program in which the format was rigid, but in which the program data were easily replaceable, first appeared in such microcomputer games as The Pinball Construction Set, by Bill Budge. This idea is easily adapted to CAI software. Such a program affords educators in all fields a means by which they can control the amount and content of drill and test sequences without having to learn programming themselves. The Editor program allows the instructor to develop unlimited sequences of lessons containing 25 chords each. The program has

been designed so that it is unnecessary for the instructor to have programming skills in order to enter lessons.

2. THE NEED FOR MUSIC THEORY CAI

In the past, there has been a great deal of emphasis on spelling and recognizing chords, but due to the time involved in composing and grading tests, and the impracticality of students learning the material without supervision, there has rarely been enough practice given in this area. CAI programs fulfill both these functions extremely well. The content is controlled by the instructor, but the program handles testing, grading, and record keeping. Moreover, the students receive instant feedback on their answers. Since the program can randomize problems, making the possibility of a repeated test unlikely, students can practice on a given set of lessons until a satisfactory performance level has been achieved. CAI has been successfully applied in aural training. The strategy should work equally well when applied to spelling and recognizing chords.

3. PROJECT DESCRIPTION

The purpose of this study was to develop programs to give drill and tests on chord spelling and visual identification. The programs were to be menu-driven for ease of use, and were to allow the instructor full control over the amount and content of course material. The programs were designed to be used as support material for the music theory sequence. The programs were written in the Applesoft

BASIC microcomputer language, but also access various utility and machine language subroutines for speed and record-keeping purposes.

The project consists of three separate programs, each of which fulfills a specific function; an Editor program, a Writechords program, and a Namechords program. The Editor program is for use only by the instructor, and is contained on a separate disk. It allows the instructor to create, modify, or delete lessons. The Writechords program gives drill and tests on spelling chords. The Namechords program gives drill and tests on naming chords. Both programs are for use by the students, and utilize the lessons which have been created using the Editor program. All the programs use standard chord-naming conventions for common-practice period harmonic theory.

CHAPTER I

A REVIEW OF THE RELATED LITERATURE

1. THE DEVELOPMENT OF MICROCOMPUTER-BASED VISUAL MUSIC INSTRUCTION

The computer has had a far-reaching impact on music instruction. It has been extensively used for ear-training purposes since 1967 when Deihl[1] introduced computer-assisted instruction for instrumental musicians at the University of Pennsylvania. This was a program which compared student responses to visual and aural versions of musical examples containing different rhythms, phrasing, and articulation. Tests administered to participating students before and after use of the program indicated a significant increase in performance and aural discrimination. The musical examples were taped, and the tape recorder was controlled by an IBM 1500 computer. In 1973 Gooch[2] developed computer-generated sound at the University of Illinois on the system known by the acronym of PLATO. The advent of computer-generated sound gave the system immediate access to musical

---

[1]John M. Eddins, "A Brief History of Computer-Assisted Instruction in Music." College Music Symposium, Vol. 21, No. 2 (Fall 1981), pp. 7-14.

[2]G. David Peters, "Hardware Development for Computer Based Instruction." College Music Symposium, Vol. 21, No. 2 (Fall 1981), pp. 15-21.

examples, thus guaranteeing instantaneous responses to student input. This system was adapted for ear-training by Peters[3] and Placek.

These initial studies set the standard for computer-based music instruction (CBMI) as a multi-user aural discrimination method designed to run on large computers. As late as 1981 Eddins writes that "there is also a growing interest in developing stand-alone student stations controlled by microcomputers which use programs developed on a large computer system. Most of these systems are too exploratory to be developed yet, but there are some exceptions."[4]

The PLATO system was expanded by Placek[5] at the University of Georgia, to include visual theory fundamentals such as not identification, and the writing of close position chords in bass or treble clef. Hofstetter,[6] at the University of Delaware, developed the GUIDO system, which is based on PLATO, to teach some aspects of harmony as well as ear-training. Ottman[7] describes the comprehensive ear-training system developed at North Texas State University.

---

[3]G. David Peters, "Capabilities of Computer-assisted Instruction in Music: The PLATO Music Project." Paper presented at the Conference of the Association for the Development of Computer-based Instructional Systems, Dallas, Texas, March 1, 1978.

[4]Eddins.

[5]Robert W. Placek, "A Model for Integrating Computer-Assisted Instruction Materials into the Music Curriculum." Journal of Computer Based Instruction, Vol. 6, No. 3. (February 1980), pp. 99-105.

[6]Fred T. Hofstetter, "Computer-Based Aural Training: The GUIDO System." Journal of Computer-Based Instruction, Vol. 7, No. 3. (February 1981), pp. 84-92.

[7]Robert W. Ottman, et al. "Development of a Concept-centered Ear-training CAI System." Journal of Computer-Based Instruction, Vol. 6, No. 3 (February 1980), pp. 79-86.

These and other systems now in use incorporate four teaching strategies: (1) tutorial, where the system teaches new concepts; (2) simulation, where the student and the computer interact in a learning environment; (3) gaming, where the student competes with other students or with computer records; and (4) drill-and-practice, where the student increases his/her abilities with concepts presented in the classroom. The advantages that CBMI has over traditional methods are these: the computer never tires, and, if correctly programmed, never makes errors; the student receives instant feedback and correction, and the computer can generate data to inform the instructor of each student's progress; the computer can randomly select test examples from a large pool of questions, thus guaranteeing individualized instruction, and accurate test results; and it increases practice time by being available when the student is ready to work.[8]

The large computers provide a great deal of computing power, but are too expensive for most music departments. There has been a great deal of work done in CBMI using low-cost Apple microcomputers. The use of small computers can cut the cost of CBMI implementation down to a reasonable size. The Apple II microcomputer is the most used microcomputer in CBMI. It is relatively inexpensive, its 6502 processor is powerful, and it was one of the first machines with easily programmable high-resolution graphics capable of music

---

[8]E. Paul Torrance, "Implications of Whole-Brained Theories of Learning and thinking for Computer-Based Instruction." Journal of Computer-Based Instruction, Vol. 7, No. 4, (May 1981), pp. 99-105.

notation. Boody[9] in 1977 began a research project to develop ear-training programs for microcomputers using the Apple II. The Apple Company, and Williams and Schrader[10] at Illinois State University soon followed suit. Some of these programs are commercially available.

The educational software that has appeared has several shortcomings. In a 1982 survey, Fisher[11] found that most software is of poor quality, and is not easily adaptable to different teaching methods or course content. The problem seems to be that researchers in CBMI applications are uninterested in producing commercial products, while the software companies are producing products that are not state-of-the-art.[12] This situation is being addressed at several schools, including Florida State University,[13] Illinois State University,[14] and The University of Tennessee.[15] Dr. Donald Pederson

---

[9]Nan Watanabe, "Review of Audio Interfacing Literature for Computer-Assisted Music Instruction." Journal of Computer-Based Instruction, Vol. 6, No. 3 (February 1980), pp. 87-90.

[10]Ibid.

[11]Francis D. Fisher, "Computer-Assisted Education: What's Not Happening?" Journal of Computer-Based Instruction, Vol. 9, No. 1, (Summer 1982), pp. 19-27.

[12]Ibid.

[13]Jack A. Taylor, "Professional Report: CBI Research and Development Centers--Center for Music Research, Florida State University." Journal of Computer-Based Instruction, Vol. 9, No. 4, (Spring 1983), pp. 171-172.

[14]G. David Peters, "Hardware Development for Computer Based Instruction." College Music Symposium, Vol. 21, No. 2, (Fall 1981), pp. 15-21.

[15]Gerald Woodyard Chastain, "A Study to Develop and Implement a Computer-Based Melodic Interval Course and to Compare that Course with an Established Drill Method." Thesis, The University of Tennessee, Knoxville, 1980.

at The University of Tennessee has developed a program in which an electronic keyboard is used as a mechanism for entering the notes of melodic dictation drills and tests. This program is designed to run on Apple II+ or IIe computers, and has been in use since 1984.[16] It is one of a set of programs which are being developed as support material for the full range of ear-training and harmonic theory courses.

The use and progress of CBMI has been aided by the National Consortium for Computer-Based Musical Instruction, formed in 1975 to promote dissemination of information among researchers in the field. The Association for the Development of Computer-Based Instructional Systems (ADCIS), publishes the Journal of Computer-Based Instruction, which devotes an annual issue to CBMI.

2. COMPUTER-ASSISTED INSTRUCTION IN MUSIC THEORY

The rapid growth of programs in music theory to assist with mastery of aural skills has not been paralleled in the area of visual skills. The difficulties inherent in graphics programming, and, until recently, the small amount of memory space in microcomputers have combined to make these types of programs either impossible, or too limited for advanced students.

---

[16]Dr. Donald Pederson, Melodic Dictation, a computer program to give drill and practice on melodic dictation. Presently in use at The University of Tennessee.

The GUIDO system,[17] and the MEDICI system[18] both incorporate music graphics in their ear-training coursework, but neither is specifically designed to test visual theory skills, such as chord identification. Several programs have been written to test visual identification of intervals, including one by Chastain[19] at The University of Tennessee. A voice-leading drill has been developed by Conrad[20] at Indiana University, but students must type the note names, which necessarily makes the program less realistic. The first program to teach visual recognition of chords was developed as a joint project between the University of Minnesota and the University of Nebraska-Omaha. The program contained twenty drills and two tests consisting of multiple choice questions, or those which require single keypress answers. The results of the initial year of use showed a dramatic improvement in student understanding of music fundamentals, as well as a much reduced attrition rate.[21] A program to teach 4 voice chord recognition developed by Apple Computer Corporation is currently in use at the University of Illinois. The program does not

---

[17]Hofstetter.

[18]Steven R. Newcomb, Bradley K. Weage, and Peter Spencer. "MEDICI: Tutorial in Melodic Dictation." Journal of Computer-Based Instruction, Vol. 7, No. 3, (February 1981), pp. 63-69.

[19]Chastain.

[20]Dr. Gary E. Wittlich, "Developments in Computer-Based Music Instruction and Research at Indiana University." Journal of Computer-Based Instruction. Vol. 6, No. 3, (February 1980), pp. 62-71.

[21]Dorothy Gross, and Roger E. Folta. "Ideas on Implementation and Evaluation of a Music CAI Project." College Music Symposium, Vol. 21, No. 2, (Fall 1981), pp. 22-26.

teach chord spelling. LASSO, a program to teach 16th century counterpoint, was developed by Newcomb in 1984.[22] This is probably the most ambitious theory program to date.

The use of computers for music theory requires graphics programming that allows the student to interact with the computer in a manner which simulates the traditional pencil and staff paper. Requiring the student to type note names promotes unintentional mistakes. Materials should be simple to use, have data files that are easily changed by an instructor unfamiliar with programming, and flexible terminology. In addition, the author feels that future efforts should be directed at developing microcomputer courseware which is suitable for students at all levels, and affordable for both schools and private teachers.

---

[22]S. R. Newcomb, "LASSO: A Computer-Based Tutorial in 16th Century Counterpoint." Dis Abst 44:3622A-3A, June, 1984.

CHAPTER II

PROJECT DEVELOPMENT

1. PROJECT OBJECTIVE

The primary objective of this project was to provide instructors with a useful alternative to written quizzes and tests on chord identification and spelling. The advantages of CAI in the development of aural skills have been proven. It is time to develop serious applications in visual music theory.

Teaching Strategy

The students will learn the theory of chord identification and spelling in class. The computer programs have been designed to reinforce the class presentation by allowing the students to practice identifying and spelling selected sets of chords. The programs provide immediate corrective feedback. Since the problems are presented in random order, the lessons can be used many times. The programs provide the students and the instructor with a set of grades for the material which accurately reflects the students' ability to spell and identify chords correctly.

2. STUDENT PREREQUISITES

These programs were designed to give drill and tests to students already familiar with the basics of common-practice harmonic

theory. There is no attempt made to teach this theory within the programs. Students will gain familiarity with the naming conventions, and learn to use the skills they have been taught in class. It is hoped that by using these programs they will gain speed and accuracy so that analysis of musical scores will be made easier and more practical. Students must be familiar with the process of naming triads and seventh chords before these programs become practical; however, the course content can be controlled so that the lessons appear in a graded sequence, i.e., root position triads and seventh chords appear before inversions and augmented sixth chords.

3. MENU-DRIVEN INPUT

The programs have all been designed so that no programming experience is required for their use. They are all menu-driven. Each time the computer requires input from the student or the instructor, a series of choices is presented, much like a restaurant menu. The student or instructor either types a number or points an arrow to obtain the desired choice. Arrow control is achieved by using the left and right arrow keys in connection with the space bar on the Apple keyboard. This method ensures that only expected inputs are obtained by the computer, and absolves the user from the need for typing skill, and memorizing complicated rules for using the programs. When typing is required, the program checks the input for proper format, and gives an appropriate message if an error is discovered.

4. PROGRAM OUTPUT

The program uses a cathode ray tube (CRT) monitor screen as a template for writing chords and chord names. All graphics, both text and musical notation, are in high-resolution mode and drawn in black on a white background. A color monitor is desirable but not necessary.

Data Storage

The programs and student data are stored in magnetic form on two 5-1/4 inch floppy disks. One contains the Editor program and the master copy of the lessons, and the other contains the Namechords and Writechords programs, and the students' copies of the lessons.

5. PROGRAM CONTROL

The content and amount of course material is controlled by the instructor using the Editor program. The students are presented with the choice of using either the Namechords or the Writechords program. They are allowed to choose from up to 5 lessons. A lesson is chosen by pressing an identifying number, and then submenus offer choices for naming the chords, or the student can write chords by moving the arrow to the correct line or space for each note, and then pressing the space bar. The programs request a social security number and a first name from each student before they begin work. This ensures that students will work on the correct set of lessons, as the programs will not let them sign on unless they are correctly registered

with the record-keeping system. The Editor works in the same fashion, but no sign-on routine is required. The instructor can create lessons or access any previously saved lesson by choosing one of several activities from a menu. The option to quit and return to the menu is nearly always available in all three programs.

6. RECORD KEEPING

Student files are kept for each student on the reverse side of the floppy disk containing the Writechords and Namechords programs. Each record contains a social security number, first and last name, and the high score for each of the five lessons available to that student in each quarter of work. The records are stored in three identical utility files, one for each quarter. Lesson files and the template graphics for the programs are also stored on the reverse side. The Editor program saves lessons on the same side of the floppy disk where the program is located. These must be transferred to the student disks using the Applesoft DOS System Master FID program.

CHAPTER III

PROJECT DESIGN

1. THE EDITOR PROGRAM

The Editor program is the central processor for lessons. It is designed for access by the instructor to create, modify or delete lessons. It is almost completely menu-driven for ease of use, and consistency of data input.

The Main Menu

The main menu appears on the screen when the Editor disk is booted. The menu allows the instructor to choose one of 5 activities or quit.

1. VIEW PROBLEMS
2. ERASE PROBLEMS
3. ADD PROBLEMS
4. DELETE LESSON
5. CREATE LESSON
6. QUIT

Program flow is determined by the activity selected. At the end of each activity, the program returns to the main menu.

View Problems

When the view problems activity is selected, the program will ask for a lesson name. The lesson desired is loaded into memory, and the instructor can step through all problems, or view any individual problem. Problems can be viewed by the following keypresses. The right and left arrow keys respectively increase or decrease the problem number by one each time they are pressed. The problem appears on the screen when the <return> key is pressed. The routine begins with problem 0, so no problem appears on the screen initially. Any individual problem can be viewed by pressing the associated problem number and the <return> key. Since lessons may only contain 25 chords, the maximum problem number allowed is 25, and the minimum is 1. The chord on the screen is erased whenever the problem number is changed, and the <return> key is pressed.

The lesson names are kept in a file called LESSON NAME. Each time a lesson name is required by the program, this file is checked. If the lesson does not exist, the available lessons are printed on the screen, and the program asks for a lesson name again.

The escape key <ESC> is always an option. If the <ESC> key is pressed, the program returns to the main menu.

Erase Problems

The erase problems routine functions in exactly the same manner as the view problems routine, but adds one keypress option. If the <E> key is pressed while a problem is on the screen, the program will ask the instructor if that problem should be erased.

If the answer is yes, the problem is deleted from the lesson. If the answer is no, the program returns to the view routine. This allows the instructor to step through lessons, and erase any problems which are incorrect, or which are unsuitable for the course. When the <ESC> key is pressed, the instructor is asked if the lesson should be saved. If the answer is yes, the problems are packed together, and saved under a chosen name and course number. Lesson names may not exceed 25 characters, and must begin with a letter or number. The Writechords and Namechords programs are designed to use lessons with 25 problems. If there are fewer than 25 problems in a lesson, the programs will ask for nonexistent information. The instructor should not erase problems without adding replacements immediately. After the lesson is saved, the program returns to the main menu. If the instructor has erased several problems from a lesson, there is a considerable wait (approximately 1 minute) before the program returns to the main menu.

Add Problems

The add problems subroutine allows the instructor to replace erased problems, or to add problems to unfinished lessons. A problem contains two parts, the chord as musical notation, and the chord name, in written symbols. At the beginning of the routine, the template graphics appear on the screen, the problem number appears at the lower right corner, and an arrow appears on the right side of the grand staff. This arrow points to lines or spaces, and is the control mechanism for entering notes. The arrow is controlled by pressing the left and right arrow keys on the Apple keyboard. Each time a key is

pressed, the arrow on the screen moves up or down one line or space. A note may be placed on the screen by pressing the space bar. When a note has been entered, the arrow on the screen appears under a menu containing a flat, a natural, and a sharp, and the word "erase." The keypress sequence for moving the screen arrow, and choosing the desired selection is the same as that for selecting notes. If a sharp or flat is selected, the appropriate symbol is placed on the screen to the left of the notehead. Natural signs do not appear on the screen. Selecting "erase" will erase the notehead, and return the screen arrow to the right of the staff.

This sequence is repeated until four notes have been placed on the screen, with appropriate accidentals, after which the program continues to the subroutine which requests the chord name. If a three-note chord is desired, the <return> key may be pressed after the third note has been entered, and the program continues to the subroutine which requests the chord name.

A note may be erased at any time by moving the screen arrow till it points at that note, and pressing the space bar. If this occurs, the program produces a "beep," and the screen arrow appears under the word "erase." At this point, an accidental may be added, deleted or changed, or the note and accidental (if any) may be erased by the appropriate selection.

After the chord is complete, the program allows one more chance to change notes by asking if the chord is "O.K." If the answer is yes, the program continues to the subroutine which requests the chord

name. If not, the screen arrow reappears to the right of the staff, and changes can be made using the sequence described above.

The second part of the add problems subroutine allows the instructor to enter chord names. After a chord has been entered on the staff, the program will request a chord name in the following manner:

1.) root letter name

2.) accidental for the letter name

3.) quality of the chord

4.) inversion

Each request is answered by the instructor using the arrow control mechanism described above. If the quality of the chord is that of an augmented-sixth chord, the program requests one of three types, and the letter name of the tonic of that chord, followed by a request for the accidental of that tonic letter name.

The routine continues until the lesson contains 25 chords, or until the <ESC> key is pressed. In either case, the instructor is asked if the lesson should be saved, and a course number and name for the lesson are requested if the answer is affirmative. The lesson is saved and the program returns to the main menu.

Delete Lesson

The delete lesson subroutine is used by the instructor to remove lessons from the Editor disk. This activity should be used with care, as lessons that are deleted are unrecoverable. The program requests a lesson name, and deletes the lesson from the Editor disk.

Lessons deleted from the Editor disk should also be deleted from the student disks using the Applesoft System Master FID program. All lessons begin with "LSN." followed by the lesson name. Each lesson has three parts, differentiated by the suffixes "N", "V" or "P". Lesson files with the suffix "P" contain the code which represents the problem name. Lesson files with the suffix "V" contain the code which represents the vertical position of the notehead on the screen. Lesson files with the suffix "N" contain the code which allows the program to decipher the note name, regardless of its screen position.

Create Lesson

This subroutine works in the same manner as the add problems subroutine, but no lesson name is requested by the program until the <ESC> key is pressed, or until 25 problems have been entered.

Quit

This choice returns the instructor to keyboard control of the computer. The memory is not cleared, and the program may be run again if desired.

2. PROBLEM PARAMETERS

Problems may contain only three or four notes. All common-practice harmonic chords may be entered. No double-sharps or double-flats are allowed. The grand staff template has a range of four octaves--C to c3. Two middle C's are available between the bass and treble clefs, but no notes are available on the bass clef above middle C or on the treble clef below middle C. The author suggests

that different sets of lessons be created for use by the Namechords and the Writechords programs. The Namechords lessons should be visually varied, in both open and close position, while the Writechords lessons should consist of close position chords, which are more easily checked for note content.

The available chord qualities are: Major, Minor, Dominant, Diminished, Half-diminished, Augmented, and Augmented Sixth. These qualities are represented on the screen by symbols or abbreviations whenever possible.

The available positions include root position and all of the inversions for triads and seventh chords. Augmented sixth chord inversions are not requested, so these must appear in sounding root position in lessons used with the Writechords program.

Utilities

The Editor program uses several support utilities which are contained in separate files on the Editor disk. The HELLO program loads these utilities when the disk is booted. A complete list of these utilities may be found in Appendix B.

3. THE WRITECHORDS PROGRAM

The Writechords program is intended for student use in drill and testing of chord spelling. The program uses lesson files created by the Editor program. These files are contained on the reverse side of the Writechords and Namechords disk.

Program Flow

The program is completely menu-driven. The student is presented with the choice of running either the Writechords or the Namechords program. When the Writechords program is selected, the student is asked to sign on using a social security number and a first name. The program will search the student files at this point, and select the appropriate record. If the student record is found, a lesson menu appears on the screen. The menu presents a choice of 5 lessons to the student. After a lesson is selected, the appropriate files (3 files) are loaded from the disk. The first problem will appear on the screen. Problems consist of chord names placed under the template grand staff. The student is expected to enter the notes of the chord. No particular order is required. Answers may be placed in either or both clefs, in open or close position. The student is allowed to erase any note. If a 3 note chord is desired, the <return> key should be pressed after the third note has been entered. The program allows one final chance to change a chord after the fourth note has been entered, or after the <return> key has been pressed. The entry mechanism for the notes and accidentals is the same as has been described in the Add Problems routine in the section of this chapter dealing with the Editor program. When the student has indicated that the chord is finished, the program checks for correct inversion, and correct notes. Only five keys are normally active, the left and right arrow keys, and the <ESC>, space bar and <return> keys. The numeric keys are active when the lesson

menu is on the screen. A lesson may be stopped at any time by pressing the <ESC> key, which returns the program to the main menu.

Error Messages

There are four possibilities after a student entry: (1) if the student enters a correct response--no message is given; (2) if the student enters the correct notes, but the wrong inversion--the correct version is drawn to the right of the student entry, and the message, "correct notes, wrong inversion," is written on the screen; (3) if the student gives the correct bass note, but the wrong upper notes--the program draws the correct version, and the message "wrong notes" is written on the screen; (4) if the student gives all wrong notes--the correct version is drawn, and the messages "wrong notes" and "wrong inversion" are written on the screen. A student can determine the types of errors he/she has made by comparing the response to a correct version of the chord from the lesson entered by the instructor using the Editor program. There is more than one correct placement of a chord on the grand staff, so the placement of the original version and the placement of the student response may differ. The author suggests that for ease of response comparison, the instructor should place the chords in close position for lessons intended for use only by the Writechords program. The time needed to evaluate a student response and enter the appropriate error message is approximately one second. The program will continue to the next problem when the space bar is pressed. If a student exits an unfinished lesson by

pressing the <ESC> key, the program reminds the student that the lesson was not finished, and returns to the main menu.

4. THE NAMECHORDS PROGRAM

The Namechords program is intended for student use in drill and testing of chord recognition. The program is completely menu-driven, and uses lessons created by the instructor using the Editor program. These lesson files are located on the reverse side of the disk containing the Namechords and Writechords programs.

Program Flow

The initial sign-on and main menu are identical to those of the Writechords program described in this chapter. Problems consist of chords written on a grand staff in whole notes. The student is expected to give the root letter name and accidental, the quality, and the inversion. Special information must be given for augmented sixth chords. The entry mechanism for chord names is the same as that in the Add Problems section of the Editor program described in this chapter. After the student has finished, a chance to change the response is allowed. When the student is satisfied with the chord name, the response is evaluated. The student may exit and return to the main menu at any time by pressing the <ESC> key.

Error Messages

There is no attempt made to differentiate types of errors in the Namechords program. A response is judged either correct or incorrect. A correct response elicits a "correct" message from the

computer. An incorrect response elicits an "incorrect" message, an arrow and the words "correct answer." After a wait of approximately two seconds, the incorrect student response is replaced by the correct answer. The time required by the computer to evaluate student responses is negligible, so the delay has been programmed to allow the user to understand what is happening. This delay can be adjusted easily. (See Appendix A) The student may continue with the next problem by pressing the space bar.

The author suggests that problems and lessons intended for use by the Namechords program be placed in varying positions on the staff. This will ensure a realistic simulation of problems the student will encounter in analysis of actual music.

Record Keeping

The Namechords and Writechords programs both use a utility to keep scores for completed lessons. Scores for unfinished lessons are not saved. The instructor must enter the students in the data base using the utility editor supplied with the data base. Records should be entered in the following manner: (1) Social Security number; (2) Name, last, first; (3) initial scores should be entered as "0". The programs will automatically update the scores when a student has finished a lesson. Lessons consist of 20 problems chosen at random from the 25 problems in the lesson file. Scoring is on a scale of 0 to 100, with each problem worth 5 points. Scores are not saved for unfinished lessons. Easy access to student records is available to the instructor.

CHAPTER IV

SUGGESTED IMPLEMENTATION AND RECOMMENDATIONS

1. IMPLEMENTATION

The programs described in this study are intended for use in basic music theory courses at the college level. Since there is no attempt made to teach concepts, the programs are intended as support material only. The Editor program should be used to create lessons in a manner consistent with normal course practices. The records created by student work should be analyzed carefully for consistent types of errors. Lessons should be altered or replaced if they prove to be unsuitable.

The lessons used by the Writechords and Namechords programs should, ideally, be created separately. The Writechords lessons should be created in close position so that errors on the part of the student can be easily detected. The Namechords lessons should be created with a great deal of variety in positioning. This will help students become proficient in recognizing chords in many different configurations, and allow them to become equally proficient in reading both bass and treble clefs.

The programs should prove to be effective in providing fast, reliable chord recognition skills to students who wish to master the basics of music theory.

2. RECOMMENDATIONS

Suggestions for Future Studies

A useful study would compare the performance of students using these programs with the performance of a control group using traditional methods. Another approach might be to study the best possible arrangement of materials within the program format. A needed, and useful study, would be one which collected data on the types of CBMI work being done at different universities, the success it has had, and the types of programs and teaching methods being used. The future of CBMI may depend on studies of this type. The advantages inherent in this type of teaching method must be publicized if school administrators are to be expected to support it.

Suggested Changes and Additions to the Programs

These programs have been structurally designed so that the routines may be used for future programs using the same format. The variables are often used only within one routine at a time to minimalize problems in moving routines. The following suggestions are arranged in order of programming difficulty.

1. Double-sharps and double-flats could be added to create additional problems. This would not increase the harmonic range of the programs, but would probably make entering problems slightly easier, as it would eliminate the need to screen problems containing these accidentals.

2. The most useful change would be to create a timed loop for the student answer. This would allow the use of timed tests and drills which would simulate a classroom testing environment in which the students must budget their time.

3. If a program to drill and test interval recognition is not available, the programs could be changed to allow only two notes to be entered, and the answer routines changed to reflect the interval names rather than the present chord names.

4. The template graphics, and the set of routines to write notes on the staff, could be used to create sets of chords for Roman Numeral analyses. The routines to choose names could be converted to Roman Numerals, and short musical examples could be created using the Editor.

5. The programs could be altered to accept jazz chords and other 20th century chord structures.

6. The record-keeping capabilities could be extended to include numbers of errors for each problem, the number of sessions and examples a student has completed, and the program flow could be adjusted so that a student could not progress until a satisfactory score level had been achieved. The addition of a hard disk would make this type of record-keeping feasible, but would entail considerable additional expense.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Chastain, Gerald Woodyard. "A Study to Develop and Implement a Computer-Based Melodic Interval Course and to Compare that Course with an Established Drill Method." Thesis, The University of Tennessee, Knoxville, 1980.

2. Eddins, John M. "A Brief History of Computer-Assisted Instruction in Music." College Music Symposium, Vol. 21, No. 2, (Fall 1981), pp. 7-14.

3. Fisher, Francis D. "Computer-Assisted Education: What's Not Happening?" Journal of Computer-Based Instruction, Vol. 9, No. 1, (Summer 1982), pp. 19-27.

4. Gross, Dorothy and Roger E. Foltz. "Ideas on Implementation and Evaluation of a Music CAI Project." College Music Symposium, Vol. 21, No. 2, (Fall 1981), pp. 22-26.

5. Hofstetter, Fred T. "Computer-Based Aural Training: The GUIDO System." Journal of Computer-Based Instruction, Vol. 7, No. 3, (February 1981), pp. 84-92.

6. Newcomb, S. R. "LASSO: A Computer-Based Tutorial in 16th Century Counterpoint." Dis Abst 44:3622A-3A, June 1984.

7. Newcomb, Steven R., Bradley, K. Weage, and Peter Spencer. "MEDICI: Tutorial in Melodic Dictation." Journal of Computer-Based Instruction, Vol. 7, No. 3, (February 1981), pp. 63-69.

8. Peters, G. David. "Capabilities of Computer-assisted Instruction in Music: The PLATO Music Project." paper presented at the Conference of the Association for the Development of Computer-based Instructional Systems, Dallas, Texas, March 1, 1978.

9. Peters, G. David. "Hardware Development for Computer Based Instruction." College Music Symposium, Vol. 21, No. 2, (Fall 1981), pp. 15-21.

10. Placek, Robert W. "A Model for Integrating Computer-Assisted Instruction Materials into the Music Curriculum." Journal of Computer-Based Instruction, Vol. 6, No. 3, (February 1980), pp. 99-105.

11. Taylor, Jack A. "Professional Report: CBI Research and Development Centers--Center for Music Research, Florida State University." Journal of Computer-Based Instruction, Vol. 9, No. 4, (Spring 1983), pp. 171-172.

12. Torrance, E. Paul. "Implications of Whole-Brained Theories of Learning and Thinking for Computer-Based Instruction." Journal of Computer-Based Instruction, Vol. 7, No. 4, (May 1981), pp. 99-105.

13. Watanabe, Nan. "Review of Audio Interfacing Literature for Computer-Assisted Music Instruction." Journal of Computer-Based Instruction, Vol. 6, No. 3, (February 1980), pp. 87-90.

14. Wittlich, Dr. Gary E. "Developments in Computer Based Music Instruction and Research at Indiana University." Journal of Computer-Based Instruction, Vol. 6, No. 3, (February 1980), pp. 62-71.

APPENDIXES

APPENDIX A

ALTERABLE STATEMENTS

There are several statements in the programs which may be changed to reflect individual course numbers, and preferences of instructors. These are listed by statement number, and the function of that statement or the type of change possible. Changes to statements other than these may result in program failure.

Editor

10076----controls the length of the string in which the course number is stored. This is presently limited to 5 alphanumeric characters.

3570-----controls the length of the lessons to be added to or created. This is presently set at 25 problems per lesson. If the lesson length is changed, the problem randomizing routine in both the Namechords and Writechords programs must be changed to reflect the new value.

Namechords

206----This is the statement which runs the Main Menu if the <ESC> key has been pressed. This statement may be changed if a different menu or program is desired.

315 and 320----These two statements are controls for the number of problems which are to be randomized. These two lines must be changed if Line 3570 has been changed in the Editor program.

2470 and 2525----These two lines contain the timing loops for the "correct" and "incorrect" messages.

3302----This line prints "Theory" and the course number on the screen.

4020----This line may be removed if scores are to be given for unfinished lessons.

Writechords

3----This statement writes the course number menu on the screen. Change the print statements to reflect the appropriate course numbers.

5, 6 and 7----These statements assign a value to the COURSNUMB$ string according to the student selection from the course number menu. These assigned values should be changed to reflect the appropriate course numbers.

91----This is the statement which runs the Main Menu if the <ESC> key has been pressed. This statement may be changed if a different menu or program is desired.

220 and 230----These two statements are controls for the number of problems which are to be randomized. These two lines must be changed if Line 3570 has been changed in the Editor program.

6490----This line prints "Theory" and the course number on the screen.

8020----This line may be removed if scores are to be given for unfinished lessons.

APPENDIX B

UTILITIES

There are several utilities used by the programs. A listing of these files, and a general explanation of their use follows.

:LOMEM

This program sets lomem to load the program above the first high-resolution page.

HIGHER TEXT MCI

This is a utility program developed by Ron and Darrell Aldritch. It generates text for use on the high resolution screen. It is enabled in the beginning of each program by a call to its location. (3072)

KNOWCHORD SHAPES

This is the shape table list for all of the high-resolution shapes used in the program. It includes the notes, accidentals, the erase message, and the arrow shapes.

HI.CLEAR

This is a machine-language routine developed by Dr. Pederson of The University of Tennessee. It is a high-speed byte transfer routine used to copy sections of high-resolution memory from one location to another. Used for erasing and replacing the template background.

]APPLE

This is a font developed using the editor of the Higher Text utility. It contains the upper and lower case alphanumeric characters used to write text on the high-resolution screen.

Main Menu

This program allows students to transfer between the Write-chords and the Namechords programs without having to load programs from the Applesoft editor.

Startup

This program loads all the utilities described above. It is run directly from the HELLO program when the disk is booted.

VITA

A. Russell Jones III was born in Oak Ridge, Tennessee on April 4, 1954. He has been a resident of Knoxville, Tennessee since he was 5 years old. He attended West Hills elementary school, and graduated from Bearden High School in 1971. That summer, he entered The University of Tennessee as a pre-med major. He left school after three years to pursue a career as Head Reptile Keeper at the Knoxville Zoological Park. In 1975, he began regular performance at area restaurants as a guitarist and singer-songwriter. He began to study piano at the age of 23, and returned to The University of Tennessee in 1979 to study piano literature. He studied piano with Dr. William Carter and Dr. David Northington, and received his Bachelor of Music degree from The University of Tennessee in 1983.

He accepted an offer of a teaching assistantship in 1983, and began study toward a Master's degree under Dr. Donald Pederson at The University of Tennessee. He received the Master of Music degree with a major in Music Theory in August 1985.

The author is presently performing professionally in Knoxville, and has accepted an assistantship at the University of Illinois, to study toward a Doctoral degree in music education.