

NO-H192 422

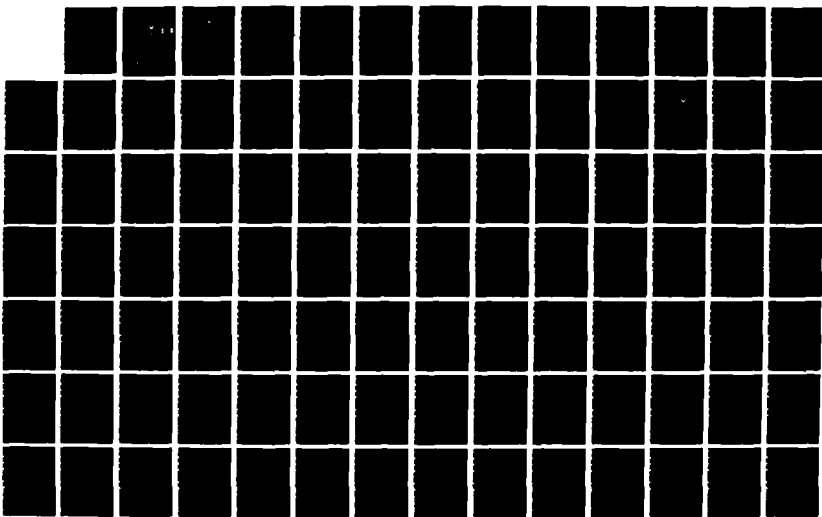
STUDENT MIX SOFTWARE SYSTEM (SMSS) REHOST(U) AIR
COMMAND AND STAFF COLL MAXWELL AFB AL
A G DECELLES ET AL APR 88 ACSC-88-0700

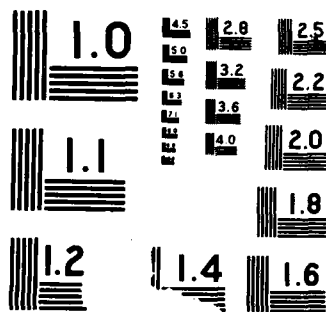
1/2

UNCLASSIFIED

F/G 12/5

NL





AD-A192 422



DTIC
ELECTE
MAY 11 1988
S D
CD

AIR COMMAND
AND
STAFF COLLEGE

STUDENT REPORT

STUDENT MIX SOFTWARE SYSTEM (SMSS) REHOST

MAJOR ARTHUR G. DECELLES

88-0700

MAJOR RICHARD M. JENSEN

"insights into tomorrow"

DISTRIBUTION STATEMENT A

Approved for public release
Distribution Unlimited

88 5 10 282



REPORT NUMBER 88-0700
TITLE STUDENT MIX SOFTWARE SYSTEM (SMSS) REHOST

AUTHOR(S) MAJOR ARTHUR G. DECELLES, USAF
MAJOR RICHARD M. JENSEN, USAF

FACULTY ADVISOR MAJOR JOHN P. BUCKNER, ACSC/DO

SPONSOR MAJOR JOHN P. BUCKNER, ACSC/DO

Submitted to the faculty in partial fulfillment of
requirements for graduation.

AIR COMMAND AND STAFF COLLEGE
AIR UNIVERSITY
MAXWELL AFB, AL 36112-5542

A192422

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT STATEMENT A Approved for public release; Distribution is unlimited.		
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) 88-0700			5. MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION ACSC/EDC		6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION		
6c. ADDRESS (City, State, and ZIP Code) Maxwell AFB AL 36112-5542			7b. ADDRESS (City, State, and ZIP Code)		
8a. NAME OF FUNDING / SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.
11. TITLE (Include Security Classification) STUDENT MIX SOFTWARE SYSTEM (SMSS) REHOST (U)					
12. PERSONAL AUTHOR(S) Decelles, Arthur G., Major, USAF; Jensen, Richard M., Major, USAF					
13a. TYPE OF REPORT		13b. TIME COVERED FROM _____ TO _____		14. DATE OF REPORT (Year, Month, Day) 1988 April	
				15. PAGE COUNT 129	
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP			
19. ABSTRACT (Continue on reverse if necessary and identify by block number) Student mixing is integral to the operation of the Air Force Air Command and Staff College (ACSC). The concept of mixing and balancing student characteristics and backgrounds within seminars is designed to increase the exposure of each student to different experiences and expertise, and facilitate self-improvement feedback from multiple instructors. The Student Mix Software System (SMSS) is used at the school to maintain student data and, in conjunction with prioritized mixing rules, assign students to seminars. This report documents the rehost and use of SMSS for the Zenith 158 personal computer, and includes the computer code and User Manual for SMSS. The SMSS rehost was designed to take advantage of the data handling and user interface capabilities of the SMART integrated software package, and the programming capabilities of GW-BASIC. The rehosted SMSS dramatically increases balancing capability, and the over-all utility of the system. The rehosted SMSS is so successful that it is already in operation; it was used to generate the third mix student assignments for ACSC class of 1988.					
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL ACSC/EDC Maxwell AFB AL 36112-5542			22b. TELEPHONE (Include Area Code) (205) 293-2867		22c. OFFICE SYMBOL

PREFACE

This report documents the rehost and use of the Student Mix Software System (SMSS) for the Zenith 158 personal computer. SMSS is used at the Air Force Air Command and Staff College (ACSC) to maintain characteristic and background data on students and, in conjunction with prioritized mixing rules, assigns students to seminars. The rehosted SMSS dramatically improves balancing capability, and the over-all utility of the system. The rehosted SMSS is so successful that it is already in operation; it was used to generate the third mix student assignments for ACSC class of 1988.

The authors wish to publicly acknowledge the assistance and support of many people who supported the completion of this project. Special thanks and recognition is given to Major John Buckner, USAF, who patiently answered the many questions necessary to finalize the rehost products. We also thank Captain Ronald Ford, USAF, who assisted in the acquisition of the GW-BASIC compiler used in this project.

This project includes over 3,000 lines of computer source code which is listed in the SMSS User Manual (Appendix A) and is maintained on three 5 1/4" computer diskettes (Appendix B of original report only). The computer code for this project may be obtained by contacting the ACSC Mix Master, ACSC/DO, Maxwell AFB, AL 36112-5564. Requestor must supply three MS-DOS formatted, double sided, double density, 5 1/4" diskettes.

This research project is submitted to simplify and optimize a process that is key to the education of all students at ACSC, that of the seminar mix. We sincerely hope that future classes capitalize on the benefits and opportunities that a balanced seminar provides, a balance achieved as a result of the rehosted SMSS.



Accession For	
NTIS CRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

ABOUT THE AUTHOR

Major Arthur G. Decelles is from Storrs, Connecticut. He graduated from the University of Connecticut in 1972 with a Bachelor of Arts degree in Mathematics and a minor in Computer Science. In 1983 he completed his Masters degree in Business Administration with an emphasis on computer based systems management from the University of Wyoming. Major Decelles received his Air Force commission in 1975 (Officer Training School) and has held positions at Headquarters Military Airlift Command (MAC); the 90th Strategic Missile Wing, Strategic Air Command; and Headquarters Electronic Systems Division (ESD), Air Force Systems Command. He has four years experience as a computer programmer/analyst satisfying MAC's Command and Control software and Computer Simulation modeling needs. At ESD, he managed the only Air Force program dedicated to reducing the exploding costs associated with the acquisition of Mission Critical Computer Resources. Major Decelles' technical experience includes an extensive knowledge of varied computer hardware systems, and the ability to program in seven different computer languages. He is a 1983 graduate of the Air Force Squadron Officer School. Major Decelles is married to the former Rita Desautels of Willimantic, Connecticut and they have two sons, Timothy and Andrew.

Major Richard M. Jensen is from Brea, California. He attended the United States Military Academy and graduated from California State University, Fullerton with a Bachelor of Arts degree in Mathematics and a Bachelor of Science degree in Computer Science in 1973. Later, he completed a Master of Arts degree in Computer Sciences from the University of Texas at Austin. Major Jensen was commissioned in the United States Air Force in 1975 and has held positions at Headquarters, Air Training Command; Headquarters Air Force; the National Emergency Airborne Command Post (Organization of the Joint Chiefs of Staff); and the Air Force Military Personnel Center. He is a graduate of the Air Force Squadron Officer School and Air Command and Staff College. Major Jensen and his wife, Laurel, have two children, Mark and Amanda.

TABLE OF CONTENTS

Preface	iii
About the Authors	iv
List of Illustrations	vi
CHAPTER ONE - INTRODUCTION TO STUDENT MIXING	1
History of Automated Mixing	2
Purpose of this Paper	3
CHAPTER TWO - THE NEED TO REHOST SMSS	5
SMSS Defects and Deficiencies	6
Benefits of Rehosting SMSS	8
CHAPTER THREE - CONCLUSION	11
Third Mix Comparisons	11
Recommended Changes	14
BIBLIOGRAPHY	17
APPENDICES:	
Appendix A - SMSS User's Manual	19
Appendix B - SMSS Computer Software (Three 5 1/4" Diskettes)	21

LIST OF ILLUSTRATIONS

FIGURES

FIGURE 1 -- Rehosted SMSS Third Mix Results	12
FIGURE 2 -- Original SMSS Third Mix Results	13

Chapter One

INTRODUCTION TO STUDENT MIXING

Student mixing is integral to the operation of the Air Force Air Command and Staff College (ACSC) at Maxwell AFB, Alabama. In fact, the mixing of ACSC students is directly linked to the mission of the school. The ACSC mission "... to enhance the professional knowledge, skills, and perspectives of mid-career officers for increased leadership roles in command and staff positions" (3:1) led to the establishment of several goals. Of particular importance to this paper are the school's goals to provide a forum for each student's professional contribution, and an environment for the personal and professional growth of each student. (3:1) The concept of mixing and balancing student characteristics and backgrounds within seminars is designed to achieve these goals by increasing the exposure of each student to different experiences and expertise, and facilitating self-improvement feedback from multiple Faculty Instructors (FIs). (3:3) The student mixing concept is such an integral part of the school's mission that the Director of Operations appoints a mix master for the entire school and each squadron commander assigns an FI as the focal point for student mixing requirements.

In addition to the extensive people resources assigned to the student mixing effort, the school is actually structured to facilitate the mixing concept. The school is currently made up of four squadrons, each consisting of up to eleven seminars of approximately thirteen students each. The school and squadron mixers balance the seminars by evenly assigning officers of like characteristics and backgrounds across seminars. For example, each seminar has at least one International Officer (IO) assigned, as well as one Army officer. Other types of officers, like females and minorities, are spread across seminars as evenly as possible. This balancing process occurs three times during the ACSC school year. With the exception of IOs and SOS faculty, ACSC students must be assigned to new seminars (different FIs) each mix. In addition, the number of seminar classmates with whom a student was previously assigned is minimized. To accommodate for the graduation of all the IOs at the end of the second mix, one seminar in each squadron is closed down for the third mix.

Assigning approximately 570 ACSC students into 44 seminars may seem like a trivial task, but when you add some thirty different rules and constraints to a process that occurs three times each school year the task quickly becomes complex and time-consuming. The basic idea behind the

mixing rules is to provide the widest range of experience possible to expose each student to new and different ideas, career fields, and service and country backgrounds. It is also important that each seminar contain representative experience and expertise as each major area of the ACSC curriculum is studied. Topics such as tactical, strategic and space operations and acquisition, logistics and budgeting functions are more quickly grasped and understood when first-hand knowledge is available within the seminar. Expertise within each seminar is especially desirable since the students present and lead the majority of the seminar lessons themselves. A well balanced student mix requires the constant and repetitive consideration of numerous factors, a task particularly well suited for a computer.

HISTORY OF AUTOMATED MIXING

Student mixing was first automated as the "Seminar Automated Mixer" (SAM) which ran on the mainframe Honeywell computer at Gunter AFS. The system was extremely slow requiring seven to ten days to receive the output reports. In addition, the SAM system violated most of the mixing rules and required extensive manual manipulation by the school and squadron mixers. (8:--) It took approximately two months prior to the start of the ACSC school year to input and process the student data, and produce just the initial reports. (4:2) Although a breakthrough from the totally manual method of mixing, the SAM system was terribly unresponsive and quite inefficient by today's standards. It was subsequently replaced with the current automated system, the "Student Mix Software System" (SMSS).

SMSS was designed to run on the Zenith 120 computer using the ZBASIC programming language and the CONDOR III Relational Data Base Management System (DBMS), systems which were readily available at ACSC. (4:3) This configuration brought the automated mixing capability directly into the mix master's office, and provided real-time, hands-on access. SMSS incorporated all the mixing rules, heuristics and factors needed by the school at the time, and it sharply reduced the amount of manpower required to produce a final mix from about 200 hours (4:31) to about 40 hours. (5:--) The system was severely constrained, however, by the limited memory capacity of the Z-120 (256K Bytes RAM) and the limited report formatting capability of CONDOR III and ZBASIC. SMSS was also unnecessarily constrained by a three tiered priority scheme for implementing the mixing rules; a constraint which severely handicapped the mix masters ability to properly balance student characteristics and skills automatically. (5:--) In the final analysis, SMSS was a giant step forward because it brought a much needed capability directly into the user's office. By providing initial reports within an hour of initiation SMSS was much more responsive than the old SAM system; so responsive that the mix master could repeatedly refine and rerun the mix several times a day. Unfortunately, as will be seen in the next chapter, this capability would become the most redeeming feature of a system which was plagued with problems.

PURPOSE OF THIS PAPER

The heart and soul of this project is the computer software and User's Manual attached to this paper (appendix A and B). The software provides the actual data manipulation, mixing, and report producing capabilities, while the User's Manual provides the ability to easily use and maintain the system. By far the majority of the effort went into the development of these products. The purpose of this paper is to establish the absolute need for these products, and consequently, justify the effort spent.

This chapter introduced the ACSC student mixing concept and provided some background into the development of SMSS. Chapter two assesses the need for rehosting the SMSS system by identifying the many shortcomings of SMSS and exploring the many benefits realized in rehosting the system. Lastly, chapter three provides some conclusions about the SMSS rehost effort, how well the new SMSS operates in comparison to the old system, and what improvements could be made to make it even better.

Chapter Two

THE NEED TO REHOST SMSS

SMSS was a tremendous step forward for ACSC student mixing but it definitely had its problems and limitations. By providing real-time, hands-on access and rapid turnaround, SMSS provided a responsiveness that was previously only dreamed of. But some SMSS functions just didn't work correctly and several others left a lot to be desired. These defects and deficiencies were important ingredients which fed into the decision to rehost SMSS, but the predominant reason for rehosting the SMSS system was its inherent lack of flexibility.

Like the earlier SAM system, SMSS was unable to accommodate even minor changes to the mixing process. Since the development of SMSS, two student characteristics ceased to be factors in the mixing process: that of "Communication Skills" and "Air Research Institute" students. (5:--) It wasn't difficult for SMSS to ignore these traits, but it was definitely a problem to incorporate new mixing rules into SMSS. The student characteristic of "Space Operations Experience" became a major mixing factor in 1987. The only way the mix master could make SMSS handle this new mixing requirement was to use the existing "Communication Skills" field to identify students with "Space Operations Experience." (5:--) This was a brute force method to make SMSS handle the new requirement, but the SMSS reports still reflected the "Communications" label. However, SMSS could not handle any other additional characteristics, even the brute force way. The ACSC mix master needed the capability to handle additional student mixing characteristics, especially considering the overwhelming emphasis being placed on "Joint Service Experience" in upcoming ACSC classes. This is a capability that SMSS currently lacks.

Another major change since the development of SMSS was an ACSC decision to standardize the school's computer hardware and software. In late 1986 the school made an investment of about \$225,000.00 in purchasing 65 Zenith 158 (IBM-compatible) personal computers, and acquiring the SMART integrated software package for each machine. (6:--) The Z-158 has more than twice the internal memory (640K Bytes RAM) of the Z-120 and, consequently, isn't as constrained in the number and type of computer application programs that can be used. In addition, the SMART software package provided a capability which integrated four vital computer applications within one system: wordprocessing, data base management, telecommunications, and spreadsheet. In essence, the school decided to standardize, for the foreseeable future, on the capabilities and user

interfaces inherent in these systems. To maximize the return on the school's investment in standard computer resources, the ACSC Director of Operations set an objective to get "... the faculty proficient and using the SMART software system." (2:1) However, as the ACSC mix master soon discovered, SMSS would not run on the Z-158s, and it was not compatible with the SMART data base manager. (5:—) This was an SMSS limitation that needed to be rectified.

SMSS clearly lacked the flexibility necessary to accomodate even the most minor mixing changes. This chapter will further assess the need to rehost SMSS by first identifying SMSS defects and deficiencies, and then by exploring the many benefits to be gained in rehosting the SMSS system.

SMSS DEFECTS AND DEFICIENCIES

SMSS has been used since 1986 to maintain student data, perform the student mixing, and generate the required reports. During that time, several difficulties were encountered in using SMSS. (5:—) The ACSC mix master was repeatedly interviewed to identify the problems and limitations of the current SMSS system, and to determine the existing student mixing requirements and practices. This information was further supplemented by reviewing related Air University and ACSC regulations dealing with seminar organization, rank, and procedures, and by inspecting the actual SMSS computer code and documentation. The result was the identification of several SMSS defects and deficiencies. A defect refers to a capability that doesn't work correctly, while a deficiency refers to a capability that is difficult to use or incorporate. Both defects and deficiencies have hidden costs associated with them.

The first defect of SMSS is that the mixing module incorrectly assigns Seminar Leaders (SLs) and Assistant Seminar Leaders (ASLs). The school has a very strict procedure for assigning SLs and ASLs, and the assignment rules change depending on the mix being processed. (1:1) The SMSS mixing module assigned students that were either not eligible for the position, or didn't assign students who were. The result was several hours of manual checking by the mix master to insure the SLs and ASLs were correctly assigned. (5:—) This was such a cumbersome task that, eventually, the mix master started pre-assigning students to seminars and to SL/ASL positions manually within the SMSS data base module. This one SMSS defect cost several hours of mix master time each mix, and considerably reduced the randomness and flexibility of the mixing process.

Another defect of the SMSS mixing module is the inaccurate statistical report summaries that it produced. (5:—) This defect had an indirect, yet more significant, cost associated with it. The mix master could get the correct mix statistics by generating SMSS data base reports using CONDOR III capabilities. This requires minimal effort by the mix master, but, the balancing capabilities of the SMSS mixing module strongly depends upon the student statistics it maintains. (4:58-97) Consequently, the

inaccurate statistics gave the mixing module a false indication of seminar balance, and directly affected student assignments. The result is the inability of SMSS to properly balance seminars, the prime SMSS function. Ineffective seminar balancing cost several hours of manual changes each mix by the school and squadron mixers to achieve an acceptable balance, a job that SMSS was created to do and which it should have done right by itself.

The last SMSS defect is that the mixing module periodically caused the computer to "hang-up". (8:--) This defect is an irritant to the mix master because the only way to get out of this condition is to re-start the computer and re-run the mix. (5:--) Fortunately, this condition occurs infrequently and cost very little in time and effort, however, it demonstrates the immaturity of the SMSS system. The SMSS mixing module contains recursive computer code (code that calls itself) and unstructured computer code. (4:58-97) Such programming practices lead to conditions that cause computers to "hang-up" and make it nearly impossible to maintain the system. All three of these SMSS defects need fixing, but the mix master could often tolerate them more than he could the other SMSS deficiencies.

The most significant deficiency of SMSS is the inadequate control it provides to the mix master over mixing rule priorities. SMSS was developed using a three tiered priority scheme which enabled the mix master to designate each mixing rule as being either "Always Important", "Sometimes Important", or "Not Important". (4:21) With the requirement to mix students on eighteen different characteristics, this priority scheme didn't provide the capability needed to define the relative importance of each rule during a given mix. As a result, the mix master had to adjust and re-run the mix several times to even come close to the desired results. (5:--) This deficiency caused several hours of mix master time in re-running the mix, and in manual adjustments to seminar assignments afterwards. The developers of SMSS recognized the balancing inefficiencies of the system and suggested an additional computer program be written to adjust the seminar assignments made by the mixing module. (4:32) It is absolutely ridiculous to write a computer program to do the job that the original computer program is suppose to do. The three tiered priority scheme incorporated within SMSS unnecessarily constrained the mix master, it caused the SMSS mixing module to work harder and longer than was really required, and in the end it caused poor mix balances. The seminar imbalances produced by SMSS are the direct result of an inadequate mixing rule priority scheme.

Another deficiency of SMSS is its poor report generation capability. As noted by the original developers, SMSS "... does not automatically produce the final report in the correct format." (4:32) This deficiency is attributable to the limitations of the CONDOR III DBMS which provides all the information necessary for the Student Statistical Report, but which cannot handle the required report format. SMSS also lacks the capability to provide real-time, ad hoc reports. (5:--) The ACSC mix

master is constantly called upon to provide special purpose, one-time reports, especially between mixes. Although this is a major requirement of the mix master, SMSS is deficient in the support it provides in this area.

Three of SMSS's capabilities don't work correctly and two others are deficient in providing all the capability the mix master's needs. These defects and deficiencies, along with the general inability of SMSS to handle changing conditions, led to the decision to rehost SMSS.

BENEFITS OF REHOSTING SMSS

Solving all the problems of the original SMSS system required much more than quick patches, it involved a complete rehosting and redesign of the system. SMSS needed the ability to run on the ACSC standard Z-158 computer using the SMART data handling and user interface capabilities. The SMSS mixing module needed to be rewritten to correct the current defects and to "optimize" the mix balancing capability. Last but not least, a comprehensive User's Manual needed to be developed. This was the scope of the SMSS rehost effort, an effort which would provide the mix master with enormously increased capabilities and the ability to produce the mixing products in a fraction of the time. The remainder of this section explores the benefits possible in four areas of SMSS: defects, deficiencies, flexibility, and maintainability.

Just fixing the three SMSS defects identified earlier would be strong justification for the rehost effort. Correcting the Seminar Leader and Assistant Seminar Leader assignment defect would save the time required to manually recheck or pre-assign students each mix. Likewise, fixing the SMSS mixing module statistics defect would save time by reducing the number of manual reassignments necessary to achieve an acceptable mix balance. Correcting the SMSS computer "hang-up" defect would save the time needed to re-run the mix and would help restore user confidence in the SMSS system. Fixing the three SMSS defects would save an estimated 20 hours each mix in the time and effort required by the school and squadron mixers to produce the necessary mixing products.

Correcting the SMSS deficiencies would not only save user time, but would provide capabilities comparable to the "responsiveness" provided by the original SMSS system. Replacing the three tiered priority scheme of SMSS with a ten level priority scheme would provide the mix master with a control over student mixing never thought possible. It would save the time required to re-run the mix over and over, and would "optimize" the student mixes. Providing the capability to generate correctly formatted reports directly with the SMSS data base module would save the time required for the mix master to reformat reports with a word-processor and would support the mix master in the real-time generation of adhoc reports. Fixing the SMSS deficiencies would provide the mix master with tremendous new capabilities; the capability to "optimize" the mixing process, and the

capability to produce reports as needed. These new SMSS capabilities alone would justify the SMSS rehost effort, but even greater benefits are possible.

Correcting the SMSS defects and deficiencies were major determinants in rehosting the SMSS system, but the predominant reason for rehosting SMSS to the Z-158 was to make the SMSS system more flexible, useable, and maintainable. The original SMSS is unable to handle additional student characteristics and mixing rules. One of the benefits of the rehost would be the addition of five unlabeled mixing identification fields which are ready for immediate use. SMSS would also be designed to allow even more fields to be added. Further flexibility would be attained by rehosting SMSS to IBM-compatible computer hardware (Z-158), and by providing three different user definable SMSS configurations: dual floppy disks, hard disk and floppy disk, and RAM disk with either hard or floppy disk. These features would increase the flexibility and useability of SMSS by expanding the number of computers SMSS could potentially run on.

Rehosting the SMSS to the Z-158 under the SMART integrated software package would make SMSS more useable and maintainable. The SMART data base manager would provide SMSS with a standard ACSC user interface capability, and would provide the SMSS user with an automated "help" feature for each SMSS menu option. The SMSS "help" feature would inform the user of each menu option available and would provide textual rationale for choosing an option. A critical product for the useability of the SMSS system is adequate user documentation. The SMSS rehost effort would produce a comprehensive User's Manual (Appendix A) which would thoroughly explain the design of the rehosted SMSS, and would guide the user through each step of the SMSS mixing process. The User's Manual would also be invaluable for modifying the SMSS computer code and for maintaining the system in the future. In addition, all SMSS computer code would be structured and documented to further enhance the maintainability of the system.

The rehost of SMSS corrected defective and deficient capabilities, and provided powerful, new capabilities for using and maintaining the system. The next chapter will compare the performance of the rehosted SMSS to the original system, and suggest improvements which could make it even better.

Chapter Three

CONCLUSION

Although the SMSS rehost effort required a complete redesign of the system and rewrite of the computer code, the original SMSS name was not changed. SMSS is an automated mixing concept, a concept which really didn't change with the rehost. The rehost fixed SMSS defects and deficiencies and provided additional capabilities, but the mixing concept stayed the same. In addition, the original SMSS was developed as a prototype system (4:3), a prototype which never reached full maturity. Consequently, the original name was retained.

The success of the SMSS rehost effort can be judged in its performance against the performance of the original system. The rehosted SMSS was used to generate the third mix student assignments for ACSC class 1988. The original SMSS was also run for purposes of comparison. The rest of this chapter compares the performance of the two systems and recommends improvements which can make the rehosted SMSS even better.

THIRD MIX COMPARISONS

Figures 1 and 2 show comparative results from the rehosted SMSS and the original SMSS for the third mix of ACSC class 1988. The results for only the 3821st Student Squadron are shown, but are representative of the entire school. The figures show the number of students SMSS assigned to each seminar which possess the characteristic listed on the left. Those student characteristics which are not completely balanced are underlined for clarity purposes. Prior to running this mix, the mix master assigned the highest rule priority of "9" to the following characteristics: Army; Reserves, ANG, Civilians, USMC, or Navy; Minorities; Females; and Medical, Legal, or Chaplains. The lowest rule priority of "0" represents characteristics which the mix master is not concerned with balancing. Priority "0" was assigned to the following characteristics: Line AD AF; No Degree; HQ Experience; Academy Graduate; Captains; and Bomber or Missile Experience. Comparable priorities were assigned by the mix master using the "Always", "Sometimes", and "Not Important" scheme of the original SMSS system. Figures 1 and 2 reflect the actual student assignments of the two SMSS systems, and do not contain any manual adjustments.

In every instance, the rehosted SMSS outperformed the original SMSS in balancing the third mix student characteristics. The only non-zero

SQUADRON 1 MIX STATISTICS MIX X

09-Feb-80

TOTAL

ATTRIBUTE	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1. Line AD Air Force	11	10	10	11	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
2. No advanced Degree	4	3	3	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
3. HQ (>=Majcom) Experience	8	9	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
4. Any Pilot	4	2	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5. Any Navigator	2	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
6. Anyhow Unaccompanied	3	3	3	4	3	3	3	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
7. USAFA (academy) grad	3	3	3	2	3	3	3	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
8. Army	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9. Res, ANG, Civ, USMC, or Navy	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10. Black or other	0	1	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11. Female	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12. Current captains	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13. Medical, legal, chaplain	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14. Bomber or missile exp	0	1	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15. Fighter pilot or WSO	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16. PPBS experience	3	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
17. Aco/Log experience	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
18. Space experience	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19. Strat Ops	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
20. unassigned	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21. unassigned	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22. unassigned	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23. unassigned	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FIGURE 1. Rehasted SMSS Third Mix Results.

11A STATISTICS - FOR SQUADRON 1

ATTRIBUTE	1	2	3	4	5	6	7	8	9	10
COMM SKILLS	1	0	2	0	2	2	1	1	1	1
SPBS SKILLS	3	3	2	2	2	3	2	3	3	3
TAC OPS SKILL	1	2	0	1	1	4	1	2	3	1
STRAT OPS SKILL	1	4	0	0	1	0	1	0	0	2
ACC/LOG SKILL	2	5	3	3	4	3	4	3	4	3
PILOT	5	4	3	4	3	3	4	2	1	2
NAVIGATOR	1	2	1	3	1	2	3	2	1	1
SINGLE/UNAC	2	0	0	3	2	1	1	1	1	2
USAF/GRADS	5	4	3	0	5	1	3	2	5	1
ARMY	1	1	2	1	1	1	1	1	1	1
RES/NG/USN/USMC	1	1	0	1	1	0	0	0	0	1
MINORITIES	1	1	1	0	1	1	1	0	1	1
FEMALES	1	0	0	1	1	1	1	0	1	1
RANK - CAPT	0	1	2	2	2	0	0	1	0	0
B8xx/B9xx/9xxx	0	1	0	0	1	0	1	2	1	1
SR ORG EXP	11	9	6	0	6	7	9	8	0	7
ARI/SOS	0	0	1	2	0	0	0	1	0	0
NO MASTER ED	6	2	2	3	0	4	4	4	4	3

FIGURE 2. Original SMSS Third Mix Results.

priority characteristics that the rehosted SMSS failed to perfectly balance were "Unaccompanied" and "Pilots" which were assigned the lowest two non-zero priorities for this mix. Even the "Unaccompanied" category was only off from a perfect balance by two students, and the "Pilot" category was off by one student. On the other hand, the original SMSS failed to balance six characteristics of concern, including one with the highest priority (Medical, Legal, Chaplain/88XX, 89XX, 90XX). The other unbalanced characteristics of interest are: Strat Ops; Acq/Log Exp; Pilots; Navigators; Space Exp; and Unaccompanied. In addition, the original SMSS was often off in its balance by a factor of four. Clearly, the rehosted SMSS outperformed the original system. In fact, it actually "optimized" the third mix.

In addition to optimizing the third mix, the rehosted SMSS proved many of its other capabilities. The portability of the system was demonstrated by running the system on the Z-120 computer with a Gemini Board. This hardware configuration allows the Z-120 to emulate an IBM-compatible computer. The rehosted SMSS ran at about half the speed as on the Z-158, but otherwise operated without difficulty. The user-friendly, menu-driven interface of the rehosted SMSS was demonstrated by the ability of a novice user, the 3824 STUS mixer, to operate the system with minimal assistance. The system's flexibility was demonstrated by successfully using an unlabeled characteristic field during the mixing process. The student information transferred from the original SMSS data base for "Strategic Operations Experience" was flawed. The flexibility of the additional characteristic fields enabled the mix master to quickly recover from this condition without having to manually edit each student record. Last, but not least, the new mixing module correctly assigned all the Seminar Leaders and Assistant Seminar Leaders. The rehosted SMSS clearly met all the performance objectives and exceeded the expectations of all concerned.

RECOMMENDED CHANGES

One limitation of the rehosted SMSS system became apparent during processing of the third mix. The SMSS mixing module is designed to assign the SLs and ASLs before making any other assignments, just as the original SMSS did. This gives the assignment of SLs/ASLs an unintended, artificial priority above all other mixing rules. (7:—) It also severely limits the school and squadron mixers ability to manually reassign students assigned to these positions, and requires that the student dates of rank be rechecked if any manual reassignments are made. If the assignment of SLs and ASLs were moved from the SMSS mixing module to the SMSS data base module, all students could be manually reassigned by the mix master without regard to the final determination of the SLs and ASLs. Besides providing additional capability, this change would provide even more flexibility and randomness to the mixing process.

The rehosted SMSS has dramatically reduced the workload of the school and squadron mixers. It provides much better products, with greater

flexibility, in about the same amount of time as the original system. But, regardless how good the rehosted SMSS, no automated mixing system will ever totally remove all manual adjustments. There are always person unique mixing factors, such as personality, which cannot be automated, but which will always play an important role. The rehosted SMSS "optimizes" the mixing process and, as a tool, provides a tremendous capability to the mix master; a capability which is highly flexible, useable, and maintainable, and which will continue to satisfy the school's future student mixing needs.

BIBLIOGRAPHY

A. REFERENCES CITED

Official Documents

1. Air University. Seminar Organization and Responsibilities, ACSCR 35-2, Maxwell AFB, AL. 1 September 1987.
2. Air University. ACSC SMART Survey, ACSC/DO, Air Command and Staff College, Maxwell AFB, AL. 26 January 1988.

Unpublished Materials

3. Air University. Operations Handbook, Academic Year 1988, Air Command and Staff College, Maxwell AFB, AL. 1988.
4. Ritchhart, Kenneth M., Major USAF and Simmons, Robert L., Major USAF. ACSC Project 86-2120, The Student Mix Software System (SMSS). Research Study prepared at the Air Command and Staff College, Air University, Maxwell AFB, AL. April 1986.

Other Sources

5. Buckner, John P., Major, USAF. ACSC Mix Master, Air Command and Staff College, Maxwell AFB, AL. Multiple Interviews: Oct-Dec 87.
6. Ford, Ronald D., Captain, USAF. Chief ACSC Information Systems, Air Command and Staff College, Maxwell AFB, AL. Interview: Oct 87.
7. Hansen, Steven L., Major, USAF. 3824 STUS Mixer, Air Command and Staff College, Maxwell AFB, AL. Multiple Interviews: Feb 88.
8. Romer, Irving, F., Major, USAF. Former ACSC Mix Master, Air Command and Staff College, Maxwell AFB, AL. Interview: Jan 88.

B. RELATED SOURCES

Official Documents

9. United States Air Force. Rank, Procedures, and Command, AFR 35-54, Maxwell AFB, AL. 15 September 1981.

CONTINUED

Other Sources

10. Innovative Software's SMARTWARE. The SMART Data Base Manager. PO Box 15998, Lenexa, KS. 66215-9990, 1986.
11. Innovative Software's SMARTWARE. The SMART System Manual. PO Box 15998, Lenexa, KS. 66215-9990, 1986.
12. Zenith Data Systems. Microsoft GW-BASIC Compiler User's Guide, Version 3.2. Manual: 595-3900-01, St. Joseph, Michigan, 49085, 1982.
13. Zenith Data Systems. Microsoft GW-BASIC Compiler User's Reference, Version 3.2. Manual: 595-3865-01, St. Joseph, Michigan, 49085, 1982.
14. Zenith Data Systems. Microsoft GW-BASIC Interpreter, Version 2. Manual: 595-3161-04, St. Joseph, Michigan, 49085, 1982.
15. Zenith Data Systems. Microsoft MS-DOS, Version 3. Volume I - III, 595-3705-01/02/03, St. Joseph, Michigan, 49085, 1884.

APPENDICES

Appendix A

STUDENT MIX SOFTWARE SYSTEM (SMSS)

USER'S MANUAL



AIR COMMAND AND STAFF COLLEGE

STUDENT REPORT

STUDENT MIX SOFTWARE SYSTEM

USER'S MANUAL

"insights into tomorrow"

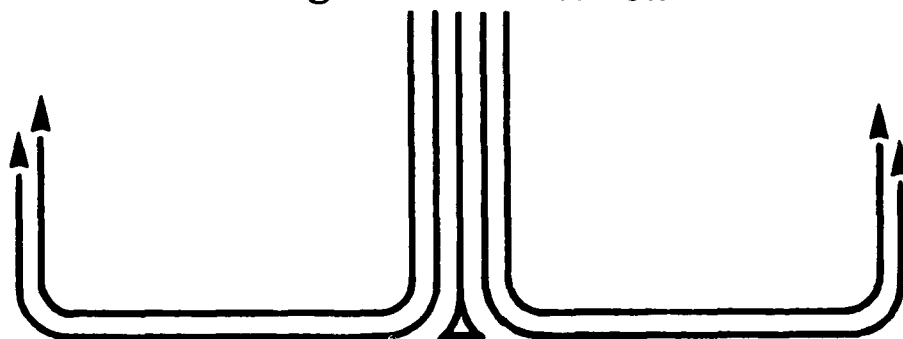


TABLE OF CONTENTS

List of Illustrations.....	iii
CHAPTER ONE -- SMSS OVERVIEW.....	1
1.1 -- What is SMSS?.....	1
1.2 -- SMSS Structure.....	1
1.3 -- Using This Manual.....	2
CHAPTER TWO -- INSTALLING AND STARTING SMSS.....	3
2.1 -- The SMSS Release Disks.....	3
2.2 -- Runtime Disk Configurations.....	3
2.3 -- Setup Steps.....	4
2.4 -- Running SMSS.....	6
2.4.1 -- Disk Option Screen.....	6
2.4.2 -- SMSS Main Menu.....	7
2.4.3 -- Menu Selection Rules.....	7
2.4.4 -- Help Screens.....	8
CHAPTER THREE -- FILE MAINTENANCE.....	11
3.1 -- Build Student Records From Another Disk.....	11
3.2 -- Add Student Records Manually.....	13
3.3 -- Edit Existing Student Record.....	14
3.4 -- Delete Student Record.....	15
3.5 -- Designate Student Numbers.....	15
3.6 -- Edit Environment Record.....	15
3.7 -- Delete International Officers.....	20
3.8 -- Post Mixer Tentative Assignments.....	20
CHAPTER FOUR -- PERFORM MIXING.....	23
4.1 -- The SMART/Mixer Interface.....	23
4.2 -- SMSS Mixing Menu.....	24
4.3 -- Mixing Sequence of Operations.....	24
CHAPTER FIVE -- PRODUCE REPORTS.....	27
5.1 -- Report Selection Menus.....	27
5.2 -- School Mixing Summary.....	29
5.3 -- Squadron Mixing Summary.....	29
5.4 -- Seminar Mixing Report.....	29
5.5 -- Complete Mixing Report.....	31
5.6 -- Alpha Rosters.....	31
CHAPTER SIX -- MIXMASTER'S CHECKLIST.....	35
APPENDICES:	
Appendix One -- Student File.....	39

Appendix Two -- SMART Project Files - Program Listings.....	43
Appendix Three -- SMSS Mixer Program - Program Listing.....	77
Appendix Four -- SMSS.BAT - Program Listing.....	101

LIST OF ILLUSTRATIONS

FIGURES

Figure 2.1 -- SMART Configure Screen.....	5
Figure 2.2 -- Disk Option Screen Display.....	7
Figure 2.3 -- SMSS Main Menu.....	8
Figure 2.4 -- Example Help Display.....	9
Figure 3.1 -- File Maintenance Menu.....	11
Figure 3.2 -- Build Student File Menu.....	12
Figure 3.3 -- Student Record Data Entry Form.....	13
Figure 3.4 -- Record Selection Menu.....	14
Figure 3.5 -- Environment Record Page 1.....	17
Figure 3.6 -- Environment Record Page 2.....	18
Figure 3.7 -- Environment Record Page 3.....	19
Figure 3.8 -- Environment Record Page 4.....	20
Figure 4.1 -- Mixing Option Menu.....	23
Figure 5.1 -- Report Mix Selection Menu.....	27
Figure 5.2 -- Reports Selection Menu.....	28
Figure 5.3 -- Report Output Selection Menu.....	29
Figure 5.4 -- School Mixing Summary.....	30
Figure 5.5 -- Squadron Mixing Summary.....	31
Figure 5.6 -- Seminar Mixing Report.....	32
Figure 5.7 -- Student Alpha Roster.....	33

Chapter One

SMSS Overview

1.1 What is SMSS?

The Student Mix Software System (SMSS) is a set of computer programs used to distribute (or mix) Air Command and Staff College (ACSC) students into ten to thirteen-person seminars to which they will be assigned for parts of the school year. The school philosophy is to distribute students evenly among seminars based on characteristics and skills of each student so that each seminar has a uniformly homogeneous "mix" of students. Students are generally reassigned three times each academic year. The following features of SMSS aid the ACSC staff in assigning students:

- Student data base.
- Menu driven program that "walks" the user through the mixing process.
- Predefined reports

SMSS is designed to run on the Zenith 158 (Z158) computer system, which is commonly used at ACSC. Additionally, it will run on any IBM-compatible personal computer with a configuration similar to the ACSC standard Z158.

1.2 SMSS Structure.

There are two main software elements to SMSS: (1) the user interface and (2) the mixing program. The user interface runs under the SMART integrated software package which is a commercial software system owned by ACSC. SMART contains a word processor, spreadsheet, data base manager, and a telecommunications package. SMSS uses the data manager to maintain its student data base, detailed in Appendix 1. Also, there is programming language within SMART, called the SMART Project Processing feature. The SMSS user interface is written in this language and runs as a SMART project file. Appendix 2 is a source listing of all the SMSS project files. See the SMART user's manuals located in the ACSC Computer Room (room 236) for more information on the SMART data manager and project processor. However, because SMSS guides the user through a menu driven process, it should not be necessary for him/her to be an expert on the SMART data manager.

SMART does not provide the detailed programming capability needed to do student mixing, so the SMSS mixing program is written in the

GW-BASIC programming language. A complete source listing of the mixer is at appendix 3. Again, a knowledge of BASIC is not needed to run SMSS.

1.3 Using This Manual.

This user's manual is organized in the same sequence as the steps involved in the student mixing process. Chapter 2 describes how to load SMSS, how to configure it for your computer, and how to get it started. Next, Chapter 3 shows you how to build and maintain the SMSS data base. Chapter 4 deals with student mixing, and Chapter 5 discusses the reports that show the results of mixing. Finally, Chapter 6 lists the sequence of steps that the ACSC "mixmaster" (the staff officer responsible for student mixing) should go through, using SMSS. Therefore, the first-time user should start at the beginning and march through sequentially as the manual will describe the mixing process as well as how to use SMSS in its support.

There are several examples throughout the manual that show sequences of commands to type into the computer. Actual keyboard inputs will be shown **boldfaced** and underlined.

Chapter Two

Installing and Starting SMSS

This chapter gives step by step instructions for installing SMSS on your computer and shows how to get the system started. If SMSS is already installed, skip to paragraph 2.4 for running instructions.

2.1 The SMSS Release Disks.

SMSS comes to you on three, 5 1/4 inch floppy diskettes. Disk 1 (SMSSPROG) contains the SMSS source and object programs, and all the auxiliary files needed to run and maintain SMSS. Because the whole system will not fit on one diskette, the large student data base is on disk 2 (SMSSSTUS). Together, disk 1 and disk 2 contain all of SMSS. A third disk (SMSSRUN) is similar to disk 1 but contains only the object, or run-time versions of the mixer and user interface programs. Disk 3 is useful if space is limited and you don't want to load up the program source code or if you want to release the system to someone else and don't wish to make the source code available. Make a backup copy of all three diskettes using the MS-DOS, "DISKCOPY" command.

2.2 Runtime Disk Configurations.

Some actions you will take to prepare your computer for SMSS depend on how you plan to run the system. SMSS can be run from either of three disk configurations using (1) floppy disk, (2) hard disk, or (3) a combination of floppy disk and virtual (in-memory) disk, sometimes called ramdisk. Each configuration has certain benefits and constraints in terms of speed and transportability as follows:

FLOPPY DISK ONLY -- If you don't have room on your hard disk to permanently install SMSS, or you want to be able to transport the system between computers, this is the way to go. You must have a computer with two floppy disk drives -- drive A will be used for the SMSSPROG or SMSSRUN disk and drive B will be used for the SMSSSTUS disk. This is the slowest-running option.

HARD DISK -- If you've got a hard disk with at least 750k bytes available, this is the fastest and preferable way to run SMSS. Transportation between computers can be tedious.

FLOPPY AND VIRTUAL DISK -- Virtual disk (VDISK) is an MS-DOS feature that allows you to set aside an area of memory that is treated as if it were a disk device. In this option, SMSS programs run from floppy drive A, but the student data base is copied into VDISK at the beginning of your session and copied back to floppy disk at the end. This method is a compromise between the other two options both in terms of speed and transportability.

If you're not constrained to a particular mode of operation, or haven't decided how you will configure SMSS, go ahead with all the following setup steps to allow all modes of operation.

2.3 Setup Steps

Load the SMSSPROG disk into drive A (the upper floppy disk drive) and SMSSSTUS into drive B (the lower one). Type cd a: to switch to drive A. Perform any of the following steps that are applicable to your configuration.

SETUP HARD DISK -- If you will run SMSS from hard disk, you must create a directory and copy all of SMSS to it. There is a routine on the SMSSPROG disk to do this. Type hardset to run this routine. HARDSET creates a subdirectory called SMSS and copies the SMSSPROG and SMSSSTUS disks to it.

SETUP SYSTEM PATH COMMAND -- SMSS is designed so that it can run from any subdirectory or disk. Therefore, SMART must be accessible from wherever SMSS is working. Type in the system command, path. The response will be a list of subdirectories which MS-DOS searches to find program names entered as commands. The directory, \smart, must be among this list. If it is not, change the system autoexec.bat file so that it includes \smart in its path command. See your MS-DOS manual for details on the path command and the autoexec.bat file.

SETUP VIRTUAL DISK -- If you are going to run SMSS with the virtual disk option described above, you must define an area of main memory to set aside for virtual disk. This is done in the system config.sys file. See the MS-DOS manual for a detailed discussion of virtual disk and the config.sys file. There is a routine on the SMSSPROG disk to create a proper config.sys. Type vdiskset to run this routine. To activate the newly defined virtual disk after vdiskset runs, you must re-boot the system by striking the "Ctrl", "Alt", & "Del" keys simultaneously (remove the floppy disks from their drives first). As long as this config.sys file is active, 180Kb of main memory will always be set aside for virtual disk. SMSS will run on the standard ACSC Z158 computer with this memory reserved, but some other very large applications may not.

SETUP CONFIG.SYS -- Whether or not you use virtual disk, your **config.sys** file must contain a **"FILES=20"** command and a **"BUFFERS=20"** command. **Vdiskset** inserts these commands for you. But, if you do not run **vdiskset** because you don't want to configure virtual disk, you must manually edit **config.sys** to insert both these commands.

```

Configuration
  Parity: Even Odd None
  Stop bits: 1 2
  Plotter pen speed (1-10): 10

----- Graphics Display Screens -----
1 IBM Color Display Adapter      6 Apricot-xi
2 AT&T 6300                      102 IBM EGA - RGB
3 IBM 3270 PC                    103 IBM EGA - enhanced/monochrome
4 Hercules                      110 Tandy 2000
5 STB Graphics Plus II(color) 201 IBM Monochrome
-----

Graphics display screen number: 1

Paging file path:  c:\smartdat

Application data paths
  Spreadsheet/Graphics:  c:\smartdat
  Word Processor:  c:\smartword
=> Data Manager:

=====
F1 Help      F2 Edit text    F3 Blank text    F10 Finished
configure

CONFIGURE - change configuration settings

```

Figure 2.1 SMART Configure Screen

SETUP SMART DATA PATH -- SMSS is designed to operate from whichever directory it is called from. For example, if you are running from floppy disk, you would first enter **cd a:** to make drive A the active directory before calling SMSS. Similarly, if you wanted to run from hard disk, you would first enter **cd \smss** to activate the SMSS subdirectory. But, on most ACSC Z158 computers, SMART is set up to look to the floppy drive A only for database data. Therefore, you must first use the SMART **"configure"** command to set up SMART

correctly. Call SMART and blank out the Data Manager application data path by keying the following commands:

<u>smart</u> (Enter)	Calls SMART main menu
<u>3</u>	Chooses command line 3
<u>c</u>	Selects the "Configure" command
<u>↓</u>	Repeat the down arrow until you see a screen display like fig 2.1
(Space) (Enter)	Hit the space bar to blank out the "Data Manager" field
<u>F10</u>	F10 function key to exit "Configure"
<u>F10</u>	F10 function key to exit main menu
<u>Q</u>	Select "Quit" command to exit SMART

Blanking the Data Manager application data path tells SMART to get data base data from whichever subdirectory is current when SMART is called. See the SMART system manual for details of the "configure" command.

2.4 Running SMSS.

To run SMSS, you must be "in" the directory where the SMSS programs reside. In other words, if you previously loaded the system to a hard disk subdirectory called SMSS, you must first use the MS-DOS "chdir", or "cd" (for change directory) command to make SMSS the current directory. Once there, simply type the command, smss. Here's an example:

<u>cd c:\smss</u> (Enter)	Changes to the hard disk subdirectory, SMSS
<u>smss</u> (Enter)	Calls SMSS

If you're running SMSS from the floppy disk, first load the SMSSPROG diskette in drive A, then type:

<u>a:</u> (Enter)	Changes to the floppy drive
<u>smss</u> (Enter)	Calls SMSS

2.4.1 Disk Option Screen

In either case, control will pass to the SMSS project file in SMART and the screen shown in figure 2.2 will appear. This screen is typical of the menu screens that appear throughout SMSS. In this one, you are asked to tell SMSS what disk option you are using: floppy, hard, or floppy with virtual disk. To select an option, move the arrow to your choice with the "↑" (up arrow) or "↓" (down arrow) keys, then press the "Enter" key. You can also press the "Esc" key to choose the default (hard disk) option and continue to the next screen. To get helpful information about this menu displayed on the screen, you can press the (upper or lower case) "H" key for help. Help screens are described in detail in paragraph 2.4.3 below.

menu level 0

WELCOME
TO
SMSS

CHOOSE DISK OPTION

==> 1. FLOPPY DISK ONLY a:(programs),b:(student file)
 2. HARD DISK c: programs & student file
 3. FLOPPY AND VDISK a:(programs),d:(student file)

*** DEFAULT (Esc) IS OPTION 2 ***

Enter h for HELP, Esc to exit to next menu

Figure 2.2 Disk Option Screen Display

2.4.2 SMSS Main Menu.

After indicating your disk configuration, You'll see some student data records flash on the screen as SMSS loads the student and other files. Note that in the lower right hand corner of the display you see messages such as, "LOADING STUDENT FILE". These are simply information messages to let you know that SMSS is performing some action and is not waiting for keyboard input. The next menu to appear is the main menu as shown in figure 2.3. Its three options: "File Maintenance", "Perform Mixing", and "Produce Reports" are major subsections of SMSS. Each is described in subsequent chapters. Every menu in SMSS looks essentially like this one and is subject to the following general menu option selection rules.

2.4.3 Menu Selection Rules.

When selecting options from an SMSS menu, the only valid keyboard inputs are the up-arrow (↑), down-arrow (↓), "Enter", "H" (upper or lower case), and "Esc" keys. All other keys will cause an audible warning beep. Here are the meanings of the valid keys:

UP OR DOWN ARROW -- Use these keys to move the display pointer up or down to the option you want to select. The pointer will wrap around if moved below the last option or above the first option.

ENTER -- Striking the "Enter" key selects whichever option the display pointer is pointing to.

ESC -- This key causes SMSS to "pop back" to the last menu displayed. "Escaping" from the main menu (figure 2.3) is the way to exit from SMSS. Exit is to the SMART data manager. From there, an "F10" and "Q" (for "quit") will exit back to MS-DOS. The one exception to the "pop back" rule is the disk option menu (figure 2.2). Escaping from that menu pops forward to the main menu.

H -- The "H" (or "h") key calls for a help screen. Each menu in SMSS has an associated help screen that explains the option selections. The next paragraph describes how to manipulate the help screens.

menu level 1

WELCOME
TO
SMSS

OPTIONS

⇒ 1. FILE MAINTENANCE

2. PERFORM MIXING

3. PRODUCE REPORTS

Enter h for HELP, Esc to exit to last menu

Figure 2.3 SMSS Main Menu

2.4.4 Help Screens.

SMSS is somewhat self-documenting since you can call up information explaining each menu option by pressing the "H" key. Figure 2.4 is an example of a help display for the main menu. If there is more than one page to the help display, move back and forth between pages with the "Pg Dn" and "Pg Up" keys. You can print the screen by holding down the "Shift" key and pressing "Prt Sc". Return to the menu with the "Esc" key. These instructions are printed at the bottom of the boxed area on each help display. Help displays are implemented in SMSS as screens of a dummy data

base called HELP. When you press the "H" key, SMSS performs a SMART "update" command on this data base. Since there is no data in HELP, the update, and hence, the SMART function key operations (F2,F3,...,F10) displayed at the bottom (below the double line) of screen are meaningless. Use only the commands defined above.

Window 1

CHOOSE DISK OPTION HELP SCREEN

page 1 of 2 HELPO

GENERAL: SMSS can be run from either of 3 different disk configurations using floppy disk, hard disk and/or virtual (in-memory) disk, sometimes called ram-disk. Each configuration has certain benefits and constrains in speed and transportability. The options are described as follows:

1. FLOPPY DISK ONLY: The SMSS system is too big to fit on one 360kB floppy diskette. In this option, the SMSSPROG diskette should be loaded in drive a: (the upper floppy drive) and the SMSSSTUS diskette, containing the student data base should be in drive b: (the lower floppy drive). You must have called SMSS from drive a:. This is the slowest option but it is very transportable from one machine to another.
2. HARD DISK: This is the fastest and preferable way to run SMSS. You must have previously created a directory on drive c: (hard or Bernoulli) and copied the SMSSPROG & SMSSSTUS diskettes to it. Also, you must have been

Pg Dn for next page; Esc to return; Shift/Prt Sc to print

Insert ON	F3 - Prev fld	F5 - Prev rec	F7 - Fld delete	F9 - Repeat fld
F2 - Date	F4 - Next fld	F6 - Next rec	F8 - Fld reform	F10 - Finished
File: help	Window: 1		Page: 1	Rec: EOF (1) Act: Y

Figure 2.4 Example Help Display

Chapter Three

File Maintenance

This chapter describes the functions that are performed in the file maintenance section of SMSS. File maintenance functions are those used to build and maintain the student and environment files. When you select FILE MAINTENANCE from the main menu, figure 3.1 will appear on the screen. The eight functions shown are described in the subsequent paragraphs. Exit the file maintenance section by pressing the "Esc" key at the file maintenance menu. Control will pass back to the main menu.

menu level 11

**FILE
MAINTENANCE**

OPTIONS

=>

1. BUILD STUDENT RECORDS FROM ANOTHER DISK
2. ADD STUDENT RECORDS MANUALLY
3. EDIT EXISTING STUDENT RECORD
4. DELETE STUDENT RECORD
5. DESIGNATE STUDENT NUMBERS
6. EDIT ENVIRONMENT RECORD (MIXING RULES)
7. DELETE IO'S (FOR MIX 3)
8. POST MIXER TENTATIVE ASSIGNMENTS

Enter h for HELP, Esc to exit to last menu

Figure 3.1 File Maintenance Menu

3.1 Build Student Records from Another Disk

This operation builds much of the student file and is generally the first SMSS function to be performed prior to the start of each academic year. The personnel office at Headquarters, Air

University (AU) can provide data on each active duty Air Force student projected into ACSC. Contact AU/DPXM at extension 6272 to request this data. They have direct access into the central Air Force personnel computer at the Air Force Military Personnel Center, but you should give them a week or two lead time. SMSS expects this data on a single file on a floppy diskette in the ASCII data format discussed in the SMART Data Base Manager Reference Guide, page Read-1. Note that only fields 1 through 27 (see Appendix One) of the student file are read in from floppy disk. All other fields are computed by the system or entered manually.

menu level 111
SMSS - BUILD STUDENT FILE
This option allows you to build the student file by reading student records from an ASCII floppy disk file. See the SMSS users manual for the correct format of this file. You can build the student file from scratch or you can add new records to the end of the existing file. *****WARNING***** If you build from scratch, any records in the current student file will be lost. OPTIONS ==> 1. BUILD STUDENT FILE FROM SCRATCH 2. ADD RECORDS TO STUDENT FILE Enter h for HELP, Esc to exit to last menu

Figure 3.2 Build Student File Menu

After you select the BUILD STUDENT RECORDS option from the SMSS file maintenance menu, you will see figure 3.2. This menu gives you the option of either totally starting from scratch or adding new student records from the floppy disk to the student file. If you choose the first (start from scratch) option, all existing records will be erased. Note -- you can use the "copystu" command described in paragraph 6.1 to save the previous data base if necessary. Next, SMSS will ask you for the name of the file on the floppy diskette. If you don't know, back out of SMSS and use the MS-DOS "dir a:" command to display the diskette directory. After one more warning message to make sure you understand what you're doing, SMSS will ask you to load your diskette into drive

A, read the diskette and build new records into the student file. If you are running SMSS from floppy disk, there will be some disk switching ordered -- just follow the directions on the screen. This operation takes several minutes and is completed when the file maintenance menu reappears.

3.2 Add Student Records Manually

Unfortunately, the whole student file cannot be built with the option of paragraph 3.1. AU/DPXM can only give us automated data on active duty Air Force students. Information on international officers, other service officers, civilians, guardsmen, and reservists must be entered manually from the keyboard. Choose this option to type in one or more student records. A data entry form like the one in figure 3.3 will appear and the cursor will move to the first field on the form. You simply type the correct information into each field. See Appendix One for the meaning and data ranges of each field. You cannot enter anything into the last column, "COMPUTED FIELDS", since these values are computed by the system based on other information in the record.

PERSONAL DATA		ACSC STUDENT RECORD		SCHOOL	SKILLS	COMPUTED FIELDS
NAME		BACKGROUND		DATA		
RANK		ED.LVL		SQUADRON	PPBS	IO
SEX		DAFSC		MIX1	ACLOG	USAF
RACE		PAFSC		MIX2	SPACE	NOMAST
MAR.STAT		2AFSC		MIX3	TRD1	SORG
SSAN		3AFSC		MIXX	T&D2	PILOT
DOB		PME1		STU.NO	TBD3	NAV
		PME2		SOS	TBD4	SINGLE
MILITARY DATA	FLYING DATA	PME3		ARI	TBD5	USAFA
COMP		MAJCOM		CC		ARMY
DOR	AERO	ORG.LVL		SRO		OTHER
PLSD	RTFD			SL		MIN
COMM	MOF			ASL		FEMALE
AC1.HRS.DT				SLX		CAPT
AC2.HRS.DT				ASLX		NONLIN
AC3.HRS.DT					COULDBSL	STRAT
						TACSK

Insert ON F3 - Prev fld F5 - Not used F7 - Fld delete F9 - Repeat fld
 F2 Date F4 - Next fld F6 - Next rec F8 - Fld reform F10 - Finished
 File: students Window: 1 Page: 1 Rec: EOF (566) Act: Y

Figure 3.3 Student Record Data Entry Form

SMSS uses the SMART "enter" command to implement this option. See the SMART Base Manager Reference Guide, page Enter-3 for the meaning of the many helpful movement and editing keys, some of which are listed at the bottom of the screen. When you are done entering records, press the "F10" function key to return to the file maintenance menu.

3.3 Edit Existing Student Record

Select this option to change any of the fields in an existing student record. You will see the menu shown in figure 3.4, which will ask you to identify a particular student record to edit. You can select a record by name, social security number (SSAN), or student number (we'll assign student numbers later). Choose one and a box will appear in which to enter the search value. You need not enter a complete name, but the SSAN or student number must be complete. SMSS will find the record and display it in a form like that of figure 3.3. If the record is not found, you'll see an error message and control will return to the file maintenance menu. If found, the SMART "update" command is used to edit the record. Use the editing keys described in the SMART Data Base Manager Reference Guide, page Enter-3. When finished, press the "F10" function key to save the changes and return to the file maintenance menu. Note that you cannot edit the NAME, SS'N, or STU.NO fields since these are data base key fields. If you must change these fields, use SMART to load the STUDENTS file with screen, STUDENT2; then "update" to edit fields and update keys.

menu level 113

STUDENT
RECORD
SELECTION

Choose which record to process by one of the following fields.

NOTE: SSAN & STUDENT NUMBER options are much faster.

OPTIONS

==> 1. NAME (or partial name)
 2. SSAN
 3. STUDENT NUMBER

Enter h for HELP, Esc to exit to last menu

Figure 3.4 Record Selection Menu

3.4 Delete Student Record

Use this option to delete a record if a student disenrolls or you enter an erroneous record. You'll see the menu shown in figure 3.4 which will ask you to identify a particular student to delete. Recommend you use SSAN or student number to be precise. If that record is not found you'll see an error message and control will pass to the file maintenance menu. If found, the record is deleted using the SMART **"delete"** command. Note that deleted records are not actually physically dropped from the student file. You can recover a deleted record by manually using the SMART **"delete"** command.

3.5 Designate Student Numbers

ACSC uses student numbers internally to identify students. Once you've created the entire student file, use this option to assign student numbers. The file is ordered alphabetically and sequential multiples of five (0005, 0010, 0015,...) are generated into the STU.NO field. If you add students after performing this operation, run it again to regenerate numbers. Once numbers are issued to students, however, you should not run this again, since everyone's assigned number will change. Note that you cannot do any mixing before student numbers are assigned because the mixing program identifies students by student number. You'll get a warning message to make sure you really want to perform this operation; then, when completed, the file maintenance menu will appear.

3.6 Edit Environment Record

SMSS maintains an environment record (in the ENVIRON file) used to pass control information to the mixing program. With this record you tell the mixer:

- Which mix (1,2 or 3) to process
- Relative priorities of mixing rules
- Organizational structure of the school

You must have edited the environment record before mixing. There are four pages to the environment record data entry form shown in figures 3.5, 3.6, 3.7, and 3.8. When you choose this option, SMSS uses the SMART **"update"** command and displays page one on the screen. Enter or change the following fields depending on the mix environment you desire:

MIX NUMBER -- The mixer uses different algorithms depending on which of the three annual mixes it is working on. Enter a value of 1, 2, or 3 here.

MAX NUMBER OF PREVIOUS SEMINAR MATES -- The mixer avoids assigning together students who have previously been assigned together in

past seminars. Enter here, the number of previously-assigned-together students you want to allow in each seminar. The number must be from 2 through 9. If the mixer is forced to violate this rule, it will issue a warning message

SYSTEM MIXING RULES AND PRIORITIES -- Students are mixed according to a set of adjustable mixing rules. The rules shown on page 1 of the environment record (figure 3.5) are system-wide rules as follows:

ARI -- Airpower Research Institute students should not change seminars. Enter 9 to honor this rule. Any other value (from 0 to 8) will be treated as a student mixing rule (see below), and ARI students will be mixed like any other student attribute.

SOS -- Squadron Officer School instructors should not change seminars. (9 to honor, any other to mix SOS instructors)

IO -- International Officers should not change seminars. (9 to honor, any other to mix IO's)

BUDDIES -- How important is it (which priority) that the number of students in the MAX-NUMBER-OF-PREVIOUS-MATES field not be assigned together? This feature is currently not implemented, and is treated as a 9 (highest) priority regardless of what you enter here.

SEMCHANGE -- All other students (not ARI, SOS, IO) must change seminars. (9 to honor, any other to ignore). This feature is not currently implemented. All other students will always change seminars.

STUDENT MIXING RULES AND PRIORITIES -- The mixing program's primary task is to distribute students equitably among seminars according to attributes and skills each student possesses. There are 23 such attributes shown in figures 3.6 and 3.7. The priorities you assign to each attribute define how the mixer will operate. It first "spreads-out" all the students with priority nine attributes, then the eights, and so on down through priority ones. Thus it is more likely that students with high priority attributes will be distributed evenly among seminars. Students with low priority attributes are more likely to "bunch up" together. For example, if the school determines that it is very important that pilots are spread out among seminars, give the PILOT attribute rule a relatively high priority number. Similarly, if it is less important that students with PPBS experience are uniformly distributed, give the PPBS attribute a relatively low priority number. The actual values from 9 down to 1 are unimportant -- only the relative values among the priorities are important. A priority of 0 (zero) means, "do not consider this

attribute at all in mixing". By experimenting with different priority relationships, you can adjust the mixer output.

The five "TBD" (To Be Determined) fields in figure 3.7 allow the system to respond to changing school mixing requirements. For example, assume that in the future, the school determines that it is important to spread out students with joint organization experience. After you identify these students, you can use the EDIT STUDENT RECORD option to mark these records with a "Y" in the TBD1 field. Then, edit the environment record to assign that attribute a mixing priority and enter "Joint Experience" in the TBD1 description field to indicate what TBD1 is being used to represent. As long as the TBD fields are not being used, their priorities should be zero.

Window 1

Used to update the single-record Environment file.

MIX
ENVIRONMENT
RECORD

Page 1 of 4

Enter MIX number... 3_ (1,2, or 3)
Enter MAX NUMBER of PREVIOUS SEMINAR MATES to mix together 3_ (2 thru 9)
Following are mix rules & priorities. Mixing will be done in priority order.
9 is the highest priority, down to 1 (lowest). Priority 0 means don't consider.

RULE	PRIORITY	DESCRIPTION
ARI	0_	ARI students don't change seminars
SOS	9_	SOS students don't change seminars
IO	0_	IOs don't change seminars
BUDDIES	8_	No more than above number of ex-mates in seminar
SEMCHANGE	9_	Must change seminar

(Experience/Background rules on next page ---hit Pg Dn)

Insert OFF F2 - Prev fld F5 - Prev rec F7 - Fld delete F9 - Repeat fld
F2 Date F4 - Next fld F6 - Next rec F8 - Fld reform F10 - Finished
File: environ Window: 1 Page: 1 Rec: EOF (1) Act: Y

Figure 3.5 Environment Record Page 1

SEMINAR STRUCTURE -- The fourth page (figure 3.8) of the environment record is used to define the organization of the

school. Enter here the first (lowest numbered) and last (highest numbered) seminar numbers in each squadron. If any seminars are missing from that range, (as they might in mix 3 after the international officers depart), you can designate up to three missing seminar numbers.

Finally, after the environment record is set like you want it, press the "F10" function key to save the changes and return to the file maintenance menu.

Window 1

MIX ENVIRONMENT RECORD

Page 2 of 4

RULE	PRIORITY	DESCRIPTION
USAF	0_	Line AD Air Force_____
NOMAST	2_	No advanced Degree_____
SORG	4_	HQ (>=Majcom) Experience_
PILOT	6_	Any Pilot_____
NAV	6_	Any Navigator_____
SINGLE	5_	Anyhow Unaccompanied_____
USafa	2_	USafa (academy) grad_____
ARMY	9_	Army_____
OTHERCOMP	8_	Res, ANG, Civ, USMC, or Navy_
MINORITY	5_	Black or other_____
FEMALE	5_	Female_____
CAPTAIN	5_	Current captains_____
NONLINE	7_	Medical, legal, chaplain_____

(More on next page..Hit Pg Dn)

Insert ON
F3 - Prev fld
F5 - Prev rec
F7 - Fld delete
F9 - Repeat fld

F2 - Date
F4 - Next fld
F6 - Next rec
F8 - Fld reform
F10 - Finished

File: environ
Window: 1
Page: 2
Rec: EOF (1)
Act: Y

Figure 3.6 Environment Record Page 2

Window 1

MIX ENVIRONMENT RECORD

Page 3 of 4

RULE	PRIORITY	DESCRIPTION
STRATOPS	9_	Bomber or missile exp_____
TACOPS	9_	Fighter pilot or WSO_____
PPBS	9_	PPBS experience_____
ACQLOG	9_	Acq/Log experience_____
SPACE	6_	Space experience_____
*****Following Can Be Defined Later if needed*****		
TBD1	0_	unassigned_____
TBD2	0_	unassigned_____
TBD3	0_	unassigned_____
TBD4	0_	unassigned_____
TBD5	0_	unassigned_____

Next Page is for designating school structure...hit Pg Dn

Insert ON

F3 - Prev fld

F5 - Prev rec

F7 - Fld delete

F9 - Repeat fld

F2 - Date

F4 - Next fld

F6 - Next rec

F8 - Fld reform

F10 - Finished

File: environ

Window: 1

Page: 3

Rec: EOF (1)

Act: Y

Figure 3.7 Environment Record Page 3

Window 1

MEDIA 1=floppy

2_ 2=hard

3=floppy/vdisk

MIX ENVIRONMENT RECORD

Page 4 of 4

SEMINAR STRUCTURE

SMSS can account for a variable number of seminars in each of 4 Squadrons. Enter the first (lowest numbered) and last (highest numbered) seminar numbers below for each squadron. Also, you can designate up to 3 missing seminars per squadron. Missing seminars must be within the first/last range.

SQUADRON	FIRST SEMINAR	LAST SEMINAR	MISSING SEMINARS (if any)
3821	1_	11	1_ 0_ 0_
3822	12	22	22 0_ 0_
3823	23	33	25 0_ 0_
3824	34	44	40 0_ 0_

*****END*****

Insert ON

F3 - Prev fld

F5 - Prev rec

F7 - Fld delete

F9 - Repeat fld

F2 - Date

F4 - Next fld

F6 - Next rec

F8 - Fld reform

F10 - Finished

File: environ

Window: 1

Page: 4

Rec: EOF (1)

Act: Y

Figure 3.8 Environment Record Page 4

3.7 Delete International Officers

International officers (IO's) graduate before the end of the academic year. Choose this option to delete IO records after they graduate. After a warning message to double check that you really want to do this, the operation is completed when you see the file maintenance menu reappear.

3.8 Post Mixer Tentative Assignments

The mixing program only makes tentative seminar assignments. Thus, after a mix, you can examine the mixing results and rerun it or make manual adjustments, if necessary. MIXX is the tentative seminar assignment field in the student file. When the mix is acceptable, use this option to move MIXX values into MIX1, MIX2, or MIX3, thereby making the mix assignment permanent. Also, the mixer sets fields SLX and ASLX to a value of "Y" if the student would be a seminar leader or assistant seminar leader, respectively, in the new mix. This option updates fields SL and

ASL based on SLX and ASLX, as well. After selecting the option, you're asked to enter a value of 1, 2, or 3 to indicate which mix to post.

Chapter Four

Perform Mixing

When the student file has been built and the environment record reflects the proper mixing arguments, you are ready to mix the students into their seminar assignments. Mixing is accomplished by selecting option 2, the PERFORM MIXING option, from the main menu of figure 2.3.

4.1 The SMART/Mixer Interface

As mentioned in Chapter One, SMART is not capable of performing the complex operations of student mixing. Therefore, SMSS contains a mixing program (the mixer) written in GW-BASIC which operates directly under MS-DOS. The mixer cannot read the student file or environment record directly since these are in a special SMART data base manager format. Therefore, we use the SMART "write" command to create two files, STU.ASC and ENV.ASC, which contain selected student and environment information in a form that is readable by the mixer. When you select the PERFORM MIXING option,

menu level 12

SMSS MIXING OPTION

If there have been no changes to the student file since the last mixing run, you can skip the ASCII file update. If there have been changes, or you don't know, be safe by running the ASCII file update.

OPTIONS

=> 1. SKIP ASCII FILE UPDATE
2. RUN ASCII FILE UPDATE

Enter h for HELP, Esc to exit to last menu

Figure 4.1 Mixing Option Menu

SMSS creates these files and passes control to the mixer, outside of SMART. The mixer then produces a results file called STUOUT.ASC and passes control again to SMART, which updates the student file with the mix results in STUOUT.ASC.

4.2 SMSS Mixing Menu

Select the PERFORM MIXING option from the main menu and you will see the two-option mixing menu shown in figure 4.1. SMSS asks you here if you want to skip the creation of the STU.ASC file. If you have mixed before and have since made no changes to the student file, you can save a little time by not writing STU.ASC since it will not have changed since the last time you mixed. In this case, select option 1, SKIP ASCII FILE UPDATE. Otherwise, select option 2, RUN ASCII FILE UPDATE.

4.3 Mixing Sequence of Operations

From this point on, mixing is essentially a "hands-off" operation that requires no user input. You can track the following operations by watching the explanatory messages at the bottom of the display screen.

1. SMART sorts the student file by rank and date of rank.
2. SMART writes ENV.ASC, the environment record interface file for the mixer.
3. If you requested it, SMART writes the student interface file, STU.ASC.
4. SMART relinquishes control and the mixer is called.
5. The mixer asks whether or not to turn off trace output. Trace output "on" will cause a trace of mixing events to be displayed on the screen. Mixing performed.
6. If there is an abort exit from the mixer, the whole process stops in MS-DOS.
7. If successful exit, control passes to SMART. SMART reads the output interface file (STUOUT.ASC) into a data manager file called NEWMIX.
8. Student file fields, MIXX, SLX, and ASLX are updated from NEWMIX using the SMART "transaction" command.
9. Control passes to the main menu.

This whole process will take several minutes, so be patient. When complete, the MIXX field of each student record will contain the tentative seminar number of assignment. SLX will contain "Y" if

the student is designated a Seminar Leader in the new mix. ASLX will be "Y" if he/she is designated as Assistant Seminar Leader. These tentative assignments are not made permanent until you perform the POST MIXER TENTATIVE ASSIGNMENTS option of paragraph 3.8.

Chapter 5

Produce Reports

SMSS produces two types of reports:

- (1) Mixing summary reports that show the results of a mix.
- (2) Alphabetically sorted student rosters.

The mixing summary reports are usually produced after y u have mixed the students. These summaries show whether you have produced a "balanced" distribution. If not, you can make manual changes to the tentative seminar assignments, or you can re-mix after adjusting mix rule priorities as in paragraph 3.6. The alpha rosters can be run at any time to support school operations.

menu level 13

SMSS REPORTS OPTION

Choose which mix to report on.

OPTIONS

==> 1. MIX1
 2. MIX2
 3. MIX3
 4. MIXX (Tentative mix results of last mixer run)

Enter h for HELP, Esc to exit to last menu

Figure 5.1 Report Mix Selection Menu

5.1 Report Selection Menus

When you select the PRODUCE REPORTS option from the main menu, figure 5.1 will appear. Here, you simply tell SMSS which mix you will be reporting on. Remember that MIXX is the tentative seminar

assignment field. Generally, you will choose MIXX when you want the results of the last mixing run. After choosing a mix, you will see the reports selection menu of figure 5.2. Choose which report you want from this menu. The rest of the paragraphs in this chapter deal with each of these report options.

menu level 131
SMSS REPORTS OPTION
OPTIONS
⇒ 1. SCHOOL MIXING SUMMARY
2. SQUADRON MIXING SUMMARY
3. SEMINAR MIXING REPORT
4. COMPLETE MIXING REPORT (all the above)
5. SCHOOL ALPHA ROSTER
6. SQUADRON ALPHA ROSTER
7. SEMINAR ALPHA ROSTER
Enter h for HELP, Esc to exit to last menu

Figure 5.2 Reports Selection Menu

The mixing summaries (options 1 and 2) each produce one page of output and are always directed to the printer. For all other reports, you will see the Report Output Selection Menu shown in figure 5.3. Choose whether you want the output sent to the printer, a disk file, or the display screen. If you choose "DISK", you will be prompted to enter a file name. Enter a single file name or complete path name, but do not include a file extension since SMART appends the extension, ".PRT", to the name you enter. This option is implemented using the SMART "report print" command. See the SMART Data Base Manager Reference Guide, page Report-25, for details. With the "DISK" option, you can later print the file you produce with the MS-DOS "print" command. Before you print, however, enter compres to put the printer in compressed text mode. COMPRES.EXE is a program supplied on the SMSSPROG disk.

menu level 1311

**REPORT OUTPUT
SELECTION**

Choose Where you want report output sent

OPTIONS

==> 1. PRINTER
2. DISK
3. SCREEN

Default (Esc) is PRINTER

Enter h for HELP, Esc to exit to next menu

Figure 5.3 Report Output Selection Menu

5.2 School Mixing Summary

This is a one-page report (figure 5.4) that totals the 23 student attributes by squadron. Each line represents a student attribute defined in the environment record. Use this report when developing mix one to determine if each squadron has a reasonable distribution of students. This report only goes to the printer.

5.3 Squadron Mixing Summary

This report (figure 5.5) is similar to the School Mixing Summary, but it displays a column for each seminar in a particular squadron. After selection, SMSS will ask for a squadron number -- Enter 1, 2, 3, or 4. Use this report to evaluate the distribution of students into seminars across a squadron. This report only goes to the printer.

5.4 Seminar Mixing Report

The Seminar Mixing Report for a single seminar is shown in figure 5.6. There is one line for each student in the seminar and each column represents one of the student mixing attributes, printed at the top of each page. A value of "1" in any column indicates that the student possesses that attribute. Students are listed in seminar leader selection order. That is, the first student who has not previously been a seminar leader (indicated by a "Y" in the SL field) would be the seminar leader in this mix. When you choose this option, SMSS asks you for the seminar(s) to report on. Your answer can be one of three options. Enter,

ALL -- to list all seminars in the school.

Sqn (where n= 1, 2, 3, or 4) -- to list all the seminars in a particular squadron.

Seminar number (e.g., 33) -- to list a single seminar.

SCHOOL MIX STATISTICS						06 Feb 88
MIX 1						
ATTRIBUTE	SQ1	SQ2	SQ3	SQ4	TOTAL	
1. Line AD Air Force	102	101	104	103	410	
2. No advanced Degree	49	46	37	41	173	
3. HQ (>=Majcom) Experience	80	78	83	76	317	
4. Any Pilot	48	52	51	50	201	
5. Any Navigator	17	17	18	14	66	
6. Anyhow Unaccompanied	29	23	34	18	104	
7. USAFA (academy) grad	25	18	22	16	81	
8. Army	11	11	11	11	44	
9. Res,ANG,Civ,USMC,or Navy	11	11	9	10	41	
10. Black or other	5	5	5	6	21	
11. Female	7	7	7	8	29	
12. Current captains	6	7	7	3	23	
13. Medical,legal,chaplain	7	5	5	5	22	
14. Bomber or missile exp	5	6	4	7	22	
15. Fighter pilot or WSO	5	8	9	8	30	
16. PPBS experience	26	34	28	22	110	
17. Acc Log experience	31	32	15	34	112	
18. Space experience	0	0	0	0	0	
19. unassigned	0	0	0	0	0	
20. unassigned	0	0	0	0	0	
21. unassigned	0	0	0	0	0	
22. unassigned	0	0	0	0	0	
23. unassigned	0	0	0	0	0	

Figure 5.4 School Mixing Summary

SQUADRON 3 MIX STATISTICS
MIX 1

06-Feb-88

ATTRIBUTE	23	24	25	26	27	28	29	30	31	32	33	TOTAL
1. Line AD Air Force	10	9	10	8	11	10	10	9	9	9	9	104
2. No advanced Degree	4	3	3	3	4	2	3	6	4	2	3	37
3. HQ (>=Majcom) Experience	8	8	9	9	10	7	6	7	7	6	6	83
4. Any Pilot	4	5	4	7	5	5	4	4	3	5	5	51
5. Any Navigator	2	2	2	1	1	1	2	2	2	1	2	18
6. Anyhow Unaccompanied	2	2	4	4	4	2	6	0	2	3	5	34
7. USAFA (academy) grad	2	3	3	1	5	2	1	1	0	3	1	22
8. Army	1	1	1	1	1	1	1	1	1	1	1	11
9. Res, ANG, Civ, USMC, or Navy	1	1	0	2	0	0	1	1	1	1	1	9
10. Black or other	0	0	1	0	1	1	0	0	0	1	1	5
11. Female	1	1	0	1	1	1	1	0	1	0	0	7
12. Current captains	0	1	2	1	0	0	0	1	0	1	1	7
13. Medical, legal, chaplain	0	0	0	1	1	0	1	0	1	1	0	5
14. Bomber or missile exp	1	1	0	0	0	0	1	0	0	0	1	4
15. Fighter pilot or WSO	1	1	1	1	2	0	1	1	1	0	0	9
16. PPBS experience	4	3	0	5	5	3	1	2	5	0	0	28
17. Acq/Log experience	2	3	1	2	2	1	1	1	1	1	0	15
18. Space experience	0	0	0	0	0	0	0	0	0	0	0	0
19. unassigned	0	0	0	0	0	0	0	0	0	0	0	0
20. unassigned	0	0	0	0	0	0	0	0	0	0	0	0
21. unassigned	0	0	0	0	0	0	0	0	0	0	0	0
22. unassigned	0	0	0	0	0	0	0	0	0	0	0	0
23. unassigned	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5.5 Squadron Mixing Summary

5.5 Complete Mixing Report

This option is an "all-the-above" report. It runs the School Summary, Squadron Summary for all squadrons, and Seminar Mixing Report for all seminars. It is convenient (although time-consuming) to run this after mixing to get a complete mixing picture. Note that if you choose "DISK" output, the two summary reports will be printed while the Seminar Mixing Report will go to disk.

5.6 Alpha Rosters

The final three report options all produce alphabetically sorted rosters similar to that shown in figure 5.7. You can create a school, squadron, seminar, or set of seminars roster. Note that these rosters contain students' name and social security number.

so they should be protected according to the privacy act. Social security numbers have been deleted from the example.

MIX STATISTICS - FOR SEMINAR 33																																	
MIX 2																																	
1) Air Force	2) No Master's					3) Sr Organ. Exp					4) Pilot					5) Navigator					6) Single Unaccompanied												
7) USAF	8) Army					9) NO/Res/Civ/Nav/MC					10) Minority					11) Female					12) Captain												
13) Nonline	14) Strat Ops Skills					15) Tac Ops Skills					16) PPBS Skills					17) Acc/Lda Skills					18) Space Skills												
19) TBD1	20) TBD2					21) TBD3					22) TBD4					23) TBD5																	
NAME	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	M1	M2	M3	RA	DOF	SL	ASL	Y	X	
BOWMAN STEVE		1						1																28	33	04	850301	Y		0	0		
STEWART JAME	1	1	1	1											1									30	33	04	850301	Y		0	0		
DALRYMPLE MI	1		1			1					1													29	33	04	850501		Y	7	0		
DOUGLAS JAME	1		1																					23	33	04	851201			0	0		
LEWIS OLIS L	1		1	1			1								1									26	33	04	860501			1	0		
TUGGLE JAMES		1							1															30	33	04	860710			0	0		
JENSEN RICH	1		1													1								31	33	27	04	860801			0	0	
DRUPPER CHAR	1		1											1		1	1							23	33	04	860901			0	0		
WILLIAMS PAU	1			1			1																	24	33	04	861001			0	0		
MIMS EDDIE R	1				1	1				1														32	33	04	870101			0	0		
DUNLAP DAVID			1			1			1				1			1								26	33	13	870606			0	0		
RANGA TEMBOR		1		1																				33	33	04	0						
AL-SHEDADI S		1		1																				33	33	04	0						
Sem Total	8	5	7	5	1	3	2	1	2	1	1	0	1	1	1	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Count	13																																

Figure 5.6 Seminar Mixing Report

***** PRIVACY ACT 1974 *****

SEMINAR 01 ALPHA ROSTER

STUDENT NUMBER	NAME	SSAN	RANK	DOR	COMMAND	PAFSC	COMPONENT	AERO RATING	MIX1	MIX2	MIX3
0020	ADAMS TIMOTHY M		04	951202	NGS	9016	ANG	NO RATING	31	27	
0025	ARMED		05	0			SOMAL	PILOT	31	31	
0070	BREAUPT SIMON D		04	951201	ELM	2825	USAF	NO RATING	31	25	
0595	DOOLEY JON D		04	850901	ATC	0950	USAF	SR NAV	31	24	
0615	DUPONT MARTIN E		04	850101	HAF	7016	USAF	NO RATING	31	29	
0610	DUTILLY MARTIN D		04	870101	ATC	2295D	USAF	SR NAV	31	26	
1045	HOIDAPL ROBERT I J		04	860401			USA		31	22	
1065	JANIED GORDON R		04	870301	PAF	5511	USAF	NO RATING	31	28	
1090	JENSEN RICHARD M		04	860901	MPC	4916	USAF	NO RATING	31	33	
2045	ROGERS ELIZABETH R		04	950701	TAC	61716	USAF	NO RATING	31	24	
2090	RUSSELL MICHAEL G		04	860101	SAC	1435V	USAF	SR PILOT	31	32	
2740	WEST GARY D		04	870301	TAC	K1115H	USAF	PILOT	31	27	

Figure 5.7 Student Alpha Roster

Chapter 6

Mixmaster's Checklist

The preceding chapters described how to use SMSS. This chapter lists the mixing tasks required of the ACSC staff officer responsible for student mixing. Numbers in parentheses are paragraph numbers of the SMSS operation to perform each task.

ACTIONS

- 6.1 After graduation in June -- Save the old student file for historical purposes. In the SMSS directory, enter copystu. This will copy all student information to floppy disk.
- 6.2 Prior to start of new class -- Get the Air Force student disk from AU/DPXM and load it. (3.1)
- 6.3 Obtain hardcopy information on non Air Force students from the ACSC Director of Operations. Manually load these students. Don't forget SOS instructors and ARI fellows. (3.2 and appendix 1)
- 6.4 When all students have been loaded -- Assign student numbers. Note that you must assign student numbers before doing any mixing. (3.5)
- 6.5 The following student attributes/skills must be entered by hand. (3.3)
- | | |
|-------------------------------|---|
| MAR.STAT | "U" for married, unaccompanied |
| SOS | "Y" for SOS phase II instructors |
| ARI | "Y" for Airpower Research Inst. fellows |
| PPBS, ACQLOG, SPACE | "Y" for possesses that skill |
- 6.6 Enter "Y" in the CC field of the class commander. (3.3)
- 6.7 Enter "Y" in SRO field of the four squadron senior ranking officers. (3.3)
- 6.8 If there are any new mixing criteria, choose TBD fields to represent them. Identify students with the attribute/skill and update appropriate TBD field with "Y". (3.3)
- 6.9 Edit all the fields in the environment record for mix 1. (3.6)

- 6.10 If you want to pre-assign any student to a particular seminar, update his/her MIX1 field with that seminar number. (3.3)
- 6.11 Perform mixing for mix 1. (4.2)
- 6.12 Run the complete mixing report on MIXX and analyze it for acceptability. (5.5)
- 6.13 If the mix is somehow "uneven", adjust the rule attribute/skill priorities and mix again until it comes out right. (3.6, 4.2)
- 6.14 To make any manual mix adjustments, edit the MIXX field. Note that if you make any manual changes, you must manually compare dates-of-rank and manually make any necessary changes to the SLX and ASLX fields. (3.3)
- 6.15 When the mix is satisfactory -- post tentative mix assignments to MIX1. (3.8)
- 6.16 Run the Alpha Rosters. (5.6)
- 6.17 Make a backup copy of the student file to protect against loss of the working copy. On a hard disk system, run "copystu" (6.1). On a floppy system, copy the SMSSSTUS disk using the MS-DOS "diskcopy" command.
- 6.18 As necessary throughout the school year -- delete any disenrolled students (3.4), and make any personnel data changes for promotions, change in marital status, etc. (3.3)
- 6.19 Prior to mix 2 -- do steps 6.8 through 6.16 for mix 2.
- 6.20 Prior to mix 3 -- Delete International Officers (3.7), then do steps 6.8 through 6.16 for mix 3.

APPENDICES

Appendix One

Student File

<u>NO.</u>	<u>NAME</u>	<u>LENGTH</u>	<u>TYPE</u>	<u>DESCRIPTION</u>	<u>RANGE</u>
(Fields 1 thru 27 come from the Air Force Personnel Data System, available from HQ AU/DPXM)					
1	NAME	27	A	Student's Name	any (e.g.) SMITH JOHN R
2	RANK	2	A	Rank/Pay Grade	01-15 (that's, zero-one)
3	SEX	1	A	Gender	M or F
4	RACE	3	A	Ethnicity	BLK, CAU or OTH
5	MAR. ST	1	A	Marital Status	D, M, S, U
6	SSAN	9	A	Social Security No.	000000000 - 999999999
7	DOB	6	N	Date of Birth	yymmdd (e.g.) 510922
8	COMP	5	A	Service Component	AFRES, ANG, CIV, USA, USAF, USMC, USG, USN or country indicator (e.g.) PAKIS
9	DOR	6	N	Date of Rank	yymmdd (e.g.) 860801
10	PLSD	6	N	Pay List Service Date (Total Service)	yymmdd (e.g.) 750815
11	COMM	7	A	Commissioning Source	ACAD MI, AF ACAD, DIRECT, DP CIV, DP MIL, OCS, OTHER, OTS, OTS DMG, ROTC, ROTC DG, ROTCDMG, UNKNOWN, USMA, USNA
12	AERO	9	A	Aeronautical Rating	MAST NAV, NAV, NAVIGATOR, NFO, NO RATING, NONE, PILOT, SR NAV, SR PILOT, (blank)
13	RTFD	4	N	Return to Fly Date	blank or yymm(e.g.) 9008

<u>NO.</u>	<u>NAME</u>	<u>LENGTH</u>	<u>TYPE</u>	<u>DESCRIPTION</u>	<u>RANGE</u>
14	MOF	3	A	Months of Flying	blank or nnn (e.g.) 085
15	AC1.HRS.DT	15	A	Most recent acft, hours, last flying date	aaaaaaaaahhhhyymm (e.g.) F-5E/F 02168701
16	AC2.HRS.DT	15	A	Next most recent aircraft, hours, date	Same as field 15
17	AC3.HRS.DT	15	A	Third most recent aircraft, hours, date	Same as field 15
18	ED.LEVEL	4	A	Education Level	BAC, BAC+, DDC, MAS, MAS+, PHD, RN, 1PDG
19	DAFSC	6	A	Duty Air Force Specialty Code	(e.g.) K1115H, or 4916
20	PAFSC	6	A	Primary AFSC	" " "
21	AFSC2	6	A	Secondary AFSC	" " "
22	AFSC3	6	A	Tertiary AFSC	" " "
23	PME1	5	A	Junior Service School Grad	(blank), ARMY, SOS, SOSC, SOSR
24	PME2	5	A	Intermediate Service School Grad	(blank), ACSC, ACSCC, ACSCS, CSC, MCCSC
25	PME3	5	A	Senior Service School Grad	(blank), MCCSC, OTHER
26	MAJCOM	3	A	Last Assigned Major Command, Special Operating Agency, or Joint Command/Agency	(blank), AAC, AAG, ACD, AFE, AFW, ATC, AUN, CBT, CMC, DMA, ELC, ELM, ESC, EUR, HAF, HRS, INT, ISC, LCT, LOG, MAC, MPC, MGS, OSI, PAF, RPC, SAC, SPC, SYS, TAC, TAP, TEC
27	ORG.LVL	3	A	Highest Organizational Level	(blank), ADV, DOD, HAF, MAJ, NAF, SOA, SQD, SQN, WNG

(Fields 28 thru 49 are entered by the mixing program or manually)

28	SQUADRON	1	A	ACSC Squadron	1, 2, 3, or 4
----	----------	---	---	---------------	---------------

<u>NO.</u>	<u>NAME</u>	<u>LENGTH</u>	<u>TYPE</u>	<u>DESCRIPTION</u>	<u>RANGE</u>
29	MIX1	2	A	Seminar Assignment for 1st Mix	01 - 44
30	MIX2	2	A	2nd Mix Seminar	" "
31	MIX3	2	A	3rd Mix Seminar	" "
32	MIXX	2	A	Tentative Mix Assigned by mixer	" "
33	STU.NO	4	A	Student Number	0005-9995
34	SOS	1	A	SOS Instructor	Y or blank
35	ARI	1	A	Air Power Research Institute Fellow	Y or blank
36	CC	1	A	Indicates Class Com- mander	Y or blank
37	SRO	1	A	Indicates Senior Ranking Officer in Squadron	Y or blank
38	SL	1	A	Has held Seminar Leader position	Y or blank
39	ASL	1	A	Has held Asst. Sem- inar leader posn.	Y or blank
40	SLX	1	A	Seminar Leader in tentative mix	Y or blank
41	ASLX	1	A	Asst. Seminar Leader in tentative mix	Y or blank
42	PPBSK	1	A	Has PPBS skill	Y or blank
43	ACQLOGSK	1	A	Has Acquisition/Log skill	Y or blank
44	SPACESK	1	A	Has Space Ops skill	Y or blank
45	TBD1	1	A	Unassigned attribute	Y or blank
46	TBD2	1	A	Unassigned attribute	Y or blank
47	TBD3	1	A	Unassigned attribute	Y or blank

<u>NO.</u>	<u>NAME</u>	<u>LENGTH</u>	<u>TYPE</u>	<u>DESCRIPTION</u>	<u>RANGE</u>
48	TBD4	1	A	Unassigned attribute	Y or blank
49	TBD5	1	A	Unassigned attribute	Y or blank

Fields 50 thru 66 are calculated fields. That is, their values (Y or blank) are calculated by the system according to formulas defined in the data base. See SMART Data Base Manager Reference Guide, page Create-9 for details. Each field has a value of blank or "Y". The "Y" condition is shown.

<u>NO.</u>	<u>NAME</u>	<u>DESCRIPTION</u>	<u>"Y" CONDITION</u>
50	IO	International Officer	COMP = anything but US components. (i.e.) not AFRES, ANG, CIV, USA, USAF, USMC, USN, USG
51	USAF	Active Air Force	COMP= "USAF" (not ANG or AFRES)
52	NOMAST	No Master's Degree	ED.LEVEL shows no Master or higher
53	SORG	Sr. Orgn Experience	ORG.LVL shows MAJCOM or higher
54	PILOT	Pilot	AERO shows pilot
55	NAVIGATOR	Navigator	AERO shows nav
56	SINGLE	No spouse	Single, divorced or unaccompanied
57	USAFA	AF Academy grad	COMM = "AF ACAD"
58	ARMY	Army Officer	COMP = "USA"
59	OTHERCOMP	Not Army or USAF	COMP = AFRES, ANG, USMC, Navy, Coast Guard or Civilian
60	MINORITY	Ethnic Minority	RACE is non-caucasian
61	FEMALE	Female	SEX = "F"
62	CAPTAIN	Rank is Captain	RANK = "03"
63	NONLINE	Not a line officer	Civilian or PAFSC is Medical, Legal or Chaplain
64	STRATOPSK	Strategic Ops skills	Missiles or Bomber background
65	TACOPSK	Tactical Ops skills	PAFSC shows Fighter background
66	COULDBSL	Qualified to be Seminar Leader	Any US officer

Appendix Two

SMART Project Files Program Listings

The following programs are written in the SMART project file programming language. See the SMART System Manual, Project Processing Guide for details.

<u>Project File</u> <u>Name</u>	<u>Page</u>	<u>Description</u>
SMSS	44	Entry and Main Routine. Displays most menus and calls other project file routines.
ENVWRITE	65	Writes environment record for mixer.
GOMIX	66	Returns to MS-DOS when mixer Requested.
IODEL	67	Deletes IO records from student file.
LOADEM	68	Loads student and help screen files.
NEWMIX	69	Called upon return from mixer. Updates student file with tentative mix results.
RAMLOAD	71	Loads student file into Virtual disk.
RAMSAVE	72	Dumps student file from virtual disk to floppy.
STUNUMB	73	Assigns student numbers.
STUREAD	74	Reads student file data from ASCII floppy disk.
STUWRITE	76	Writes student file in ASCII format for mixer.

**** Project File SMSS ****

COMMENT *****SMSS*****

COMMENT ENTRY POINT FOR MAIN SMSS PROGRAM
COMMENT WRITTEN BY MAJ RICHARD M. JENSEN

COMMENT VARIABLE & PARAMETER CONVENTIONS:
COMMENT STANDARD PROJECT VARIABLES (TEXT1,TEXT2,VALUE1,VALUE2) HAVE ONLY A
COMMENT TEMPORARY SCOPE. ALL OTHER VARIABLE NAMES (\$name) HAVE SPECIFIC
COMMENT USES AND ARE NOT USED FOR MORE THAN ONE PURPOSE.
COMMENT PARAMETERS %1-%8 HAVE TEMPORARY SCOPE. %9 AND %0 HAVE SPECIFIC USES
COMMENT (%9 INDICATES CHOSEN MIX # IN REPORTS SECTION, %0 INDICATES MEDIA
COMMENT SELECTION THROUGHOUT)

COMMENT *****
COMMENT HOUSEKEEPING AND ONE TIME SETUP

QUIET On

singlestep off

COMMENT DELETE ENV.ASC IF IT HAPPENS TO EXIST

COMMENT IN CASE WE EXIT ABNORMALLY, DON'T WANT SMSS.BAT TO THINK WE'RE MIXING
if file("env.asc")=1.0 then file erase env.asc

COMMENT OPEN THE ENVIRONMENT FILE

load environ screen environ1

COMMENT ENTRY TO SMSS FROM SMSS.BAT COULD BE FROM ONE OF 2 PLACES:

COMMENT 1. INITIAL ENTRY (PRE-MIXING) OR

COMMENT 2. AFTER MIXING

COMMENT EXISTENCE OF MIXER OUTPUT FILE, STUOUT.ASC IS INDICATOR OF WHICH.

COMMENT NOTE, MIXER CREATES STUOUT.ASC. TEST FOR STUOUT.ASC

COMMENT IF IT EXISTS GO COPY THE TENTATIVE MIXING INFO INTO THE DATA BASE

COMMENT IF NOT ASK USER WHICH MEDIA HE'S OPERATIVE FROM

if file ("stuout.asc")=1.0

execute newmix in-memory

else

call getmedia

COMMENT GETMEDIA AND NEWMIX RETURN \$MEDIA AND %0 TO INDICATE USER'S CHOICE

COMMENT OF MEDIA. DON'T USE %0 FOR ANYTHING ELSE

endif

COMMENT main menu (level 1) loop

label main

COMMENT DISPLAY WELCOME TITLE

call maintitl

let \$lvl=1

menu print 8 36 15 4 OPTIONS

menu print 10 24 15 4 1. FILE MAINTENANCE

menu print 12 24 15 4 2. PERFORM MIXING

menu print 14 24 15 4 3. PRODUCE REPORTS

COMMENT GET MENU SELECTION

\$firstline=10

**** Project File SMSS ****

```

$lastline=14
$linediff=2
$colm=23
label inloop1
call selopt
COMMENT DEBUG SINGLESTEP ON
COMMENT TEST INPUT
if $option = 1
    call preprocess
    jump main
elseif $option= 2
    call mixem
    jump main
elseif $option= 3
    call reports
    jump main
COMMENT TEST FOR H
elseif $key= 104 or $key=72
    call help
    jump main
COMMENT TEST FOR ESCAPE MEANING EXIT
elseif $key= 27
COMMENT EXISTENCE OF FILE ENV.ASC, THE ASCII ENVIRONMENT FILE, IS THE KEY
COMMENT WHETHER EXIT BACK TO DOS IS FROM HERE OR FROM A MIXING REQUEST.
COMMENT ERASE ENV.ASC TO SHOW WE'RE EXITING NORMALLY
    menu clear 15 1
    if file("env.asc")=1.0 then file erase env.asc
    unload all
COMMENT IF WE'VE BEEN WORKING ON VDISK, IT NEEDS TO BE SAVED TO FLOPPY
    if $media=3 then execute ramsave in-memory
    end
else beep on
    beep 3 illegal entry--try again
    jump inloop1
endif

```

COMMENT*****LEVEL 1 SUBROUTINES*****

```

COMMENT *****GETMEDIA*****
COMMENT ASK USER WHERE DATA IS COMING FROM--FLOPPY, HARD, OR VDISK
COMMENT OUTPUT: $MEDIA=1,2,OR 3. %O=PATH FOR STUDENTS DATABASE FILES
COMMENT NOTE ENTIRE SYSTEM IS TOO BIG FOR ONE FLOPPY DRIVE SO IF CHOICE
COMMENT IS FLOPPY, STUDENTS.* & NEWMIX GO TO B:, EVERYTHING ELSE TO A:
COMMENT ON OUTPUT, %O="B:\", "", OR "D:\
COMMENT *****
procedure getmedia
$lvl=0

```

**** Project File SMSS ****

```
label inloop0
COMMENT PUT OUT WELCOME HEADER
call maintitl
menu print 8 30 15 4 CHOOSE DISK OPTION
menu print 10 24 15 4 1. FLOPPY DISK ONLY--a: (programs), b: (student file)
menu print 11 24 15 4 2. HARD DISK          --c: programs & student file
menu print 12 24 15 4 3. FLOPPY AND VDISK--a: (programs), d: (student file)
menu print 14 22 15 4 *** DEFAULT (Esc) IS OPTION 2 ***
$firstline=10
$lastline=12
$linediff=1
$colm=23
COMMENT GET USER INPUT
call selopt
COMMENT SET DEFAULTS (IN CASE OF ESC OR 2.) TO HARD
$media=2
%0=null
COMMENT TEST FOR ESCAPE INPUT-ASSUME HARD AND EXIT
if $key=27 then jump gmend
COMMENT TEST FOR HELP REQUEST-PRINT SCREEN AND TRY AGAIN
if $key=104 or $key=72
    call help
    jump inloop0
endif
COMMENT MUST BE AN OPTION SELECTION
$media=$option
COMMENT IF HARD DISK, ALL FILES ON SMART DATAMANAGER DEFAULT CATALOG
if $option=2
    %0=null
    jump gmend
endif
COMMENT IF FLOPPY OR VDISK, NEED TO GET STUDENTS DISK IN B:
message Load STUDENTS disk into drive B:-any key when ready
COMMENT IF VDISK, GO LOAD VDISK FROM FLOPPY
if $option=3
    %0="d:\"
    execute ramload in-memory
COMMENT RAMLOAD RETURNS TEXT1=N IF USER BAILS OUT
    if text1=="n" then jump inloop0
else
COMMENT MUST BE FLOPPY, RETURN POINTING TO B:
    $media=1
    %0="b:\"
endif
label gmend
COMMENT NEED TO SAVE MEDIA IN ENVIRONMENT FILE, ELSE IT WILL GO AWAY WHEN
COMMENT WE GO OFF TO MIXER AND RETURN (i.e variable aren't saved)
goto file environ screen environl
```

**** Project File SMSS ****

```
goto record rec-number 1
let [media]=$media
COMMENT OPEN STUDENT FILE
execute loadem in-memory
return
```

```
COMMENT *****MAINTITL*****
COMMENT DISPLAY A STANDARD HEADER USED IN LEVELS 1 & 0
COMMENT *****
procedure maintitl
COMMENT PAINT SCREEN RED
menu clear 15 4
menu draw box 2 35 6 44 15 4
menu print 3 36 15 4 WELCOME
menu print 4 39 15 4 TO
menu print 5 38 15 4 SMSS
return
```

```
COMMENT*****LEVPRINT*****
COMMENT DISPLAYS THE CURRENT LEVEL NUMBER AT TOP RIGHT SCREEN
COMMENT $lvl CONTAINS LEVEL
COMMENT*****
procedure levprint
%1=$lvl
menu print 2 64 15 4 menu level %1
return
```

```
COMMENT*****FOOTER*****
COMMENT DISPLAYS THE STANDARD HELP ESCAPE LINE AT BOTTOM SCREEN
COMMENT*****
procedure footer
if $lvl=0 or $lvl=1311
    %1="next"
else
    %1="last"
endif
menu print 20 5 15 4 Enter h for HELP, Esc to exit to %1 menu
return
```

```
COMMENT *****SELOPT*****
COMMENT GIVEN A MENU, RETURNS NUMBER OF OPTION SELECTED
COMMENT *****
COMMENT INPUT $FIRSTLINE=LINE NUMBER OF FIRST OPTION
COMMENT          $LASTLINE=LINE NUMBER OF LAST OPTION
COMMENT          $COLM=COLUMN NUMBER TO PRINT > OF ==> POINTER
COMMENT          $LINEDIFF=LINE SPACING BETWEEN OPTIONS
COMMENT OUTPUT $OPTION= NUMBER OF OPTION CHOSEN
COMMENT          IF $OPTION=0, $KEY=27 FOR ESCAPE OR "H" FOR HELP
```

**** Project File SMSS ****

COMMENT *****

```

procedure selopt
$key=0
call levprint
call footer
%1=$firstline
%2=$colm-4
COMMENT POINTER SYMBOL ==>
%3=chr(205)!chr(205)!chr(16)
COMMENT DISPLAY ARROW AT 1ST OPTION
menu print %1 %2 15 4 %3
COMMENT UNTIL INPUT IS ESC, ENTER, OR h or H FOR HELP
while ($key<>27 and $key<>13 and $key<>104 and $key <>72)
COMMENT SAVE CURRENT ROW TO BLANK OUT LAST POINTER POSN
$lastrow=%1
COMMENT MOVE UP OR DOWN THE MENU OR BEEP ON BAD KEY
COMMENT CHECK FOR UPARROW (18432)
if $key=18432
    %1=%1-$linediff
COMMENT CHECK FOR DOWNARROW (20480)
elseif $key = 20480
    %1=%1+$LINEDIFF
else
    beep
endif
COMMENT WRAP POINTER IF NECESSARY
if %1 < $firstline
    %1=$lastline
endif
if %1>$lastline
    %1=$firstline
endif
COMMENT BLANK OUT LAST SPOT OF POINTER
%4=$lastrow
menu print %4 %2 4 4 " "
COMMENT PRINT POINTER AT NEW POSITION
menu print %1 %2 15 4 %3
COMMENT WAIT FOR KEY TO BE STRUCK
$key=inchar
endwhile
if $key=13
    $option=((%1-$firstline)/$linediff)+1
else
    $option=0
endif
COMMENT DEBUG lprint $option; $key
return
    
```

**** Project File SMSS ****

```
COMMENT*****HELP*****
COMMENT DISPLAYS HELP SCREEN FOR LEVEL INDICATED IN $lvl
COMMENT*****
procedure help
COMMENT CALL UP THE APPROPRIATE HELP SCREEN
%1=$lvl
load help screen help%1
paint graphics foreground 11
COMMENT THIS UPDATE IS JUST A DUMMY OPERATION TO GET THE HELP SCREENS UNDER
COMMENT USER CONTROL. NOT REALLY UPDATING ANYTHING
update
unload screen help%1
paint graphics foreground 15
goto file %0students screen student1
return
COMMENT *****PREPROCESS*****
COMMENT LEVEL 11 MENU DISPLAY
COMMENT *****
procedure preprocess
COMMENT CLEAR KEY UPDATE INDICATOR
$keyup=0
label main11
$lvl=11
COMMENT SET UP SCREEN AS USUAL
menu clear 15 4
menu draw box 2 32 5 46 15 4
menu print 3 38 15 4 FILE
menu print 4 34 15 4 MAINTENANCE
menu print 7 36 15 4 OPTIONS
menu print 9 24 15 4 1. BUILD STUDENT RECORDS FROM ANOTHER DISK
menu print 10 24 15 4 2. ADD STUDENT RECORDS MANUALLY
menu print 11 24 15 4 3. EDIT EXISTING STUDENT RECORD
menu print 12 24 15 4 4. DELETE STUDENT RECORD
menu print 13 24 15 4 5. DESIGNATE STUDENT NUMBERS
menu print 14 24 15 4 6. EDIT ENVIRONMENT RECORD (MIXING RULES)
menu print 15 24 15 4 7. DELETE 10'S (FOR MIX 3)
menu print 16 24 15 4 8. POST MIXER TENTATIVE ASSIGNMENTS
$firstline=9
$lastline=16
$linediff=1
$colm=23
label inloop11
call selopt
if $option=1
    call sturead
```

**** Project File SMSS ****

```
elseif $option=2
COMMENT LOAD SCREEN THAT ALLOWS ALL FIELD ENTRIES
  goto file %0students screen student2
  enter
  $keyup=1
COMMENT BACK TO NORMAL SCREEN
  goto file %0students screen student1

elseif $option=3
  call recfind
COMMENT TEST FOR HELP,ESCAPE OR RECORD NOT FOUND AND UPDATE IF OK
  if $option <> 0 then update

elseif $option=4
  call recfind
COMMENT TEST FOR HELP,ESCAPE,NOT FOUND,OR ALREADY DELETED AND DELETE IF OK
  if $option <> 0 and deleted=0.0 then delete

elseif $option=5
  execute stunumb in-memory

elseif $option=6
  goto file environ screen environ1
  update
  goto file %0students screen student1

elseif $option=7
  execute lodel in-memory

elseif $option=8
  menu print 15 61 15 4 Mix? (1,2 or,3)
  label ask8
  beep
  menu input 16 61 4 15 1 text1
  if text1=null then jump main11
  if text1<>"1" and text1<>"2" and text1<>"3" then jump ask8
  %1=text1
  menu print 20 35 4 15 Posting permanent mix %1
  query predefined postmix%1 neither

elseif $key=104 or $key=72
  call help

elseif $key=27
COMMENT ESCAPE-UPDATE KEYS IF ADD HAS BEEN DONE
  if $keyup<>0
  menu print 20 59 4 15 Updating key fields
  key update
```

**** Project File SMSS ****

```

        endif
    return

endif
jump main11

COMMENT *****STUREAD*****
COMMENT LEVEL 111 PROCEDURE - READ IN STUDENT RECORDS
COMMENT *****

procedure sturead
label main111
$lvl=111
COMMENT SET UP SCREEN AS USUAL
menu clear 15 4
menu draw box 2 25 4 52 15 4
menu print 3 26 15 4 SMSS - BUILD STUDENT FILE
menu print 5 15 15 4 This option allows you to build the student file by
menu print 6 15 15 4 reading student records from an ASCII floppy disk file.
menu print 7 15 15 4 See the SMSS users manual for the correct format of
menu print 8 15 15 4 this file. You can build the student file from
menu print 9 15 15 4 scratch or you can add new records to the end of the
menu print 10 15 15 4 existing file.
menu print 11 31 4 15 *****WARNING*****
menu print 12 15 15 4 If you build from scratch, any records in the current
menu print 13 15 15 4 student file will be lost.
menu print 15 36 15 4 OPTIONS
menu print 16 24 15 4 1. BUILD STUDENT FILE FROM SCRATCH
menu print 17 24 15 4 2. ADD RECORDS TO STUDENT FILE
$firstline=16
$lastline=17
$linediff=1
$colm=23
call selopt
if $option=0 and $key=27 then return
if $option=0 and ($key=104 or $key=72)
    call help
    jump main111
endif
execute sturead in-memory
return

COMMENT *****RECFIND*****
COMMENT LEVEL 113/4 PROCEDURE- FIND A PARTICULAR RECORD
COMMENT BASED ON NAME, SSAN OR STU.NO. GIVE USER THE OPTION WHICH
COMMENT *****
procedure recfind
label main113

```

**** Project File SMSS ****

```

$lvl=113
COMMENT SET UP SCREEN OF OPTIONS
call rfscreen
$firstline=11
$lastline=13
$linediff=1
call selopt
COMMENT CHECK FOR ESCAPE OR HELP AS USUAL
if $option=0 and $key=27 then return
if $option=0 and ($key=104 or $key=72)
    call help
    jump main113
endif
COMMENT INPUT A NAME, SSAN OR STU.NO ON THE OPTION LINE
COMMENT %1=SELECTED OPTION LINE NO. %2=LENGTH OF NEXT INPUT
COMMENT %3=VALUE OF INPUT, %4=KEY TO SEARCH ON, %5=FIND MODE, %6=FIND OPTION
%1=$firstline+$option-1
COMMENT ASSUME "EQUAL" AND "GLOBAL" FIND MODE AND OPTIONS
%5="equal"
%6=""
if $option=1
    %2=27
    %4="[name]"
COMMENT ON NAME SEARCH, DO (SLOWER) PART WORD AND CASE-LESS OPTIONS
    %5="partial"
    %6="iw"
    order key [name]
elseif $option=2
    %2=9
    %4="[ssan]"
    order key [ssan]
elseif $option=3
    %2=4
    %4="[stu.no]"
    order key [stu.no]
endif
COMMENT NEED TO REFRESH SCREEN SINCE ABOVE "ORDERS" DISTURBED IT
call rfscreen
menu print 10 50 15 4 ENTER SEARCH VALUE
beep
menu input %1 50 4 15 %2 text1
COMMENT GIVE ANOTHER CHANCE TO BAIL OUT
if text1 = null
    $option=0
    return
endif
menu clear 15 4
menu print 20 58 4 15 Searching for Record

```

**** Project File SMSS ****

```
%3=text1
COMMENT ALL THE ABOVE WAS SET UP TO ALLOW A BINARY (FAST) SEARCH ON SSAN OR
COMMENT STUDENT NUMBER. IT SHOULD WORK WITH "find %4 %5 %3 options g%6" BUT,
COMMENT IT DOESN'T. FOR SOME REASON I ALWAYS GET CERROR OF 3002 EVEN WHEN
COMMENT BINARY SEARCH FINDS THE RIGHT RECORD. THE FOLLOWING GETS AROUND
COMMENT THE PROBLEM BY CHECKING FOR THE ACTUAL VALUE WE'RE LOOKING FOR EVEN
COMMENT IF THE 3002 RETURN COMES BACK
find %4 %5 "%3" options g%6
COMMENT DELETED RECORDS ARE LIKE THEY'RE NOT THERE
if deleted=1.0 then jump rf2
COMMENT CERROR RETURNS 3002 IF RECORD NOT FOUND
if $option=1 and cerror<>3002 then jump rf1
text2=%4
COMMENT ON BINARY SEARCH,CHECK FOR A RETURNED VALUE THAT WE'RE LOOKING FOR
if $option<>1 and text2=text1 then jump rf1
COMMENT ELSE RECORD NOT FOUND
label rf2
    call rfscreen
COMMENT ECHO MESSAGE AND ORIGINAL VALUE
    menu print %1 6 4 15 ****NOT FOUND**
    menu print %1 50 4 15 %3
    beep 3
COMMENT SET $OPTION TO SHOW NO UPDATE ON RETURN
    $option=0
label rf1
return

COMMENT *****RFSCREEN*****
COMMENT DISPLAY THE OPTION SCREEN FOR RECFIND - LEVEL 113 MENU
COMMENT *****
procedure rfscreen
menu clear 15 4
menu print 3 35 15 4 STUDENT
menu print 4 36 15 4 RECORD
menu print 5 34 15 4 SELECTION
menu draw box 2 33 6 44 15 4
menu print 7 10 15 4 Choose which record to process by one of the following
fields
menu print 8 10 15 4 NOTE: SSAN & STUDENT NUMBER options are much faster
menu print 10 36 15 4 OPTIONS
menu print 11 24 15 4 1. NAME (or partial name)
menu print 12 24 15 4 2. SSAN
menu print 13 24 15 4 3. STUDENT NUMBER
return

COMMENT *****MIXEM*****
COMMENT LEVEL 12 PROCEDURE- CALL MIXING PROGRAM
COMMENT *****
```

**** Project File SMSS ****

```
procedure mixem
label main12
$lvl=12
COMMENT SET UP SCREEN AS USUAL
menu clear 15 4
menu draw box 2 29 4 48 15 4
menu print 3 30 15 4 SMSS MIXING OPTION
menu print 5 11 15 4 If there have been no changes to the student data base
menu print 6 11 15 4 since the last mixing run, you can skip the ASCII file
menu print 7 11 15 4 update. If there have been changes, or you don't know,
menu print 8 11 15 4 be safe by running the ASCII file update.
menu print 10 36 15 4 OPTIONS
menu print 12 24 15 4 1. SKIP ASCII FILE UPDATE
menu print 13 24 15 4 2. RUN ASCII FILE UPDATE
$firstline=12
$lastline=13
$linediff=1
$colm=23
call selopt
if $option = 0 and $key=27 then return
if $option =0 and ($key=104 or $key=72)
    call help
    jump main12
endif
COMMENT WRITE ASCII ENVIRONMENT RECORD
execute envwrite in-memory
COMMENT IF SELECTED, UPDATE ASCII STUDENT FILE
if $option=2 then execute stuwrite in-memory
COMMENT GO OFF TO MIXER UNDER DOS AND DON'T RETURN HERE
transfer gomix

COMMENT ***** REPORTS *****
COMMENT LEVEL 13 PROCEDURE - GENERATE SELECTED REPORTS
COMMENT *****
procedure reports
label main13
$lvl=13
COMMENT ASK WHICH MIX WE'RE REPORTING ON
menu clear 15 4
menu draw box 2 28 4 48 15 4
menu print 3 29 15 4 SMSS REPORTS OPTION
menu print 6 24 15 4 Choose which mix to report on.
menu print 8 36 15 4 OPTIONS
menu print 10 24 15 4 1. MIX1
menu print 11 24 15 4 2. MIX2
menu print 12 24 15 4 3. MIX3
menu print 13 24 15 4 4. MIXX (Tentative mix results of last mixer run)
```

**** Project File SMSS ****

```
$firstline=10
$lastline=13
$linediff=1
$colm=23
call selopt
if $option=0 and $key=27 then return
if $option=0 and ($key=104 or $key=72)
    call help
    jump main13
endif
COMMENT SAVE SELECTED MIX NUMBER
$mix=$option
if $mix=1
    %9="1"
elseif $mix=2
    %9="2"
elseif $mix=3
    %9="3"
else
    %9="x"
endif
COMMENT NOW LET USER SELECT REPORT
label main131
$lv1=131
menu clear 15 4
menu draw box 2 28 4 48 15 4
menu print 3 29 15 4 SMSS REPORTS OPTION
menu print 5 36 15 4 OPTIONS
menu print 7 24 15 4 1. SCHOOL MIXING SUMMARY
menu print 8 24 15 4 2. SQUADRON MIXING SUMMARY
menu print 9 24 15 4 3. SEMINAR MIXING REPORT
menu print 10 24 15 4 4. COMPLETE MIXING REPORT (all the above)
menu print 11 24 15 4 5. SCHOOL ALPHA ROSTER
menu print 12 24 15 4 6. SQUADRON ALPHA ROSTER
menu print 13 24 15 4 7. SEMINAR ALPHA ROSTER
$firstline=7
$lastline=13
$linediff=1
$colm=23
call selopt
if $option=0 and $key=27 then jump main13
if $option=0 and ($key=104 or $key=72)
    call help
    jump main131
endif
COMMENT PRODUCE REPORT
call doreport
jump main131
```

**** Project File SMSS ****

return

```
COMMENT *****DOREPORT*****
COMMENT PRODUCE REQUESTED REPORTS
COMMENT ON INPUT, $OPTION INDICATES REQUESTED REPORT
COMMENT          $MIX      INDICATES REQUESTED MIX
COMMENT *****
procedure doreport
COMMENT DETERMINE WHICH REPORT IS REQUESTED
```

```
    if $option=1
COMMENT SCHOOL SUMMARY
    call skoolsum

elseif $option=2
COMMENT SQUADRON SUMMARY-ASK WHICH SQUADRON
    menu print 7 52 15 4 Squadron? (1,2,3, or 4)
    menu print 8 52 15 4 382
    label ask2
    beep
    menu input 8 55 4 15 1 text1
    if text1=null then return
    if text1<>"1" and text1<>"2" and text1<>"3" and text1<>"4" then jump ask2
    call squadsum
```

```
elseif $option=3
COMMENT SEMINAR STAT REPORT
COMMENT SEMINAR ALPH-ASK WHICH SEMINAR(S)
    menu print 8 53 15 4 Seminar(s)?
    menu print 14 27 15 4 Enter: "ALL" for all seminars
    menu print 15 34 15 4 "SQ1","SQ2","SQ3",or "SQ4" for all in a sqdn
    menu print 16 34 15 4 Seminar number for single seminar
    label ask3
    beep
    menu input 9 53 4 15 3 text1
COMMENT PICK APPROPRIATE SEMINARS
    if text1=="all"
        jump sortem
    elseif left(text1,2)=="sq"
COMMENT SELECTING ONE WHOLE SQUADRON
        text1=mid(text1,3,1)
        menu print 20 58 4 15 Selecting Seminars
        query predefined squadron index temp2
    elseif text1=null
        return
    else
COMMENT SINGLE SEMINAR-
COMMENT MAKE SURE IT'S IN RANGE
```

**** Project File SMSS ****

```
        if val(text1)<1 or val(text1) > 50 then jump ask3
COMMENT IF HE ENTERED SINGLE DIGIT, FIX IT
        if len(text1)=1 then text1="0";text1
        menu print 20 58 4 15 Selecting Seminar
        query predefined mix%9 index temp2
        endif
        order index temp2
        label sortem
        call semsum

elseif $option=4
COMMENT COMPLETE MIXING SUMMARY-ALL SEMINARS, SCHOOL, ALL SQUADRONS
        call semsum
COMMENT CLOSE INDEX FILE TEMP
        order sequential
        call skoolsum
        $sqcnt=1
        while $sqcnt <= 4
            text1=fixed($sqcnt,0)
            call squadsum
            order sequential
            $sqcnt=$sqcnt+1
        endwhile

elseif $option=5
COMMENT SCHOOL ALPHA REPORT
        order key [name]
COMMENT DETERMINE OUTPUT MEDIUM
        call whichout
        report print alpha %4 %5

elseif $option=6
COMMENT SQUADRON ALPHA-ASK WHICH SQUADRON
        menu print 11 49 15 4 Squadron?
        menu print 12 49 15 4 382
        label ask5
        beep
        menu input 12 52 4 15 1 text1
        if text1=null then return
        %1=text1
        if text1<>"1" and text1<>"2" and text1<>"3" and text1<>"4" then jump ask5
        order key [name]
        menu print 20 58 4 15 Selecting Squadron %1
COMMENT ISOLATE THAT SQUADRON
        query predefined squadron index temp
        order index temp
COMMENT DETERMINE WHICH OUTPUT MEDIUM
        call whichout
```

**** Project File SMSS ****

```

report print alpha %4 %5

elseif $option=7
COMMENT SEMINAR ALPH-ASK WHICH SEMINAR(S)
    menu print 12 49 15 4 Seminar(s)?
    menu print 14 27 15 4 Enter: "ALL" for all seminars
    menu print 15 34 15 4 "SQ1","SQ2","SQ3",or "SQ4" for all in a sqdn
    menu print 16 34 15 4 Seminar number for single seminar
    label ask6
    beep
    menu input 13 52 4 15 3 text1
    if text1=null then return
    call getsem
COMMENT TEST FOR ERROR RETURN
    if text1="0" then jump ask6
    order index temp
COMMENT DETERMINE WHICH OUTPUT MEDIUM
    call whichout
    report print mix%9 %4 %5
endif

COMMENT MAKE SURE WE'RE LOOKING AT WHOLE FILE ON RETURN
order sequential
return

COMMENT *****SKOOLSUM*****
COMMENT PRODUCE SCHOOL MIXING STAT SUMMARY
COMMENT *****
procedure skoolsum
COMMENT SORT BY SQUADRON
    menu print 20 61 4 15 Sorting by squadron
    sort predefined squadron index temp
    order index temp
    menu print 20 47 4 15 Computing school stats
COMMENT ZERO ALL THE COUNTERS-4 SETS OF 23
    $firstunit=1
    $lastunit=4
    call czero
    goto record rec-number 1
COMMENT RUN THRU FILE-UPDATE COUNTS OF ALL ATTRIBUTES. THERE WILL BE 4
COMMENT DIFFERENT SETS OF COUNTERS FOR 4 SQUADRONS
    label nextrec1
    %1=[squadron]
    call count
    if record < records
        goto record next
        jump nextrec1
endif

```

**** Project File SMSS ****

COMMENT PRINT OUT THE REPORT WITH ALL ATTRIBUTE COUNTERS

```
menu print 20 33 4 15 Printing School Stats- Make sure printer is on
lprint
lprint repeat(" ",28);"SCHOOL MIX STATISTICS"
lprint repeat(" ",38);"MIX ";"%9";repeat(" ",25);date1(today)
lprint
lprint "    ATTRIBUTE";repeat(" ",18);"SQ1";" SQ2";" SQ3";" SQ4";" TOTAL"
lprint
value1=1
```

COMMENT THERE ARE 23 DIFFERENT ATTRIBUTES

COMMENT AND THE FIELD DESCRIPTIONS ARE IN THE ENVIRONMENT FILE

```
goto file environ screen environ1
while value1<= 23
%1=value1
```

COMMENT PRINT ATTRIBUTE COUNT, LABEL ,COUNT FOR EACH SQUADRON, & TOTAL COUNT

```
lprint repeat(" ",2-len("%1"));"%1. ";[d%1];repeat(" ",27-len([d%1]));%c%1_1;
lprint repeat(" ",4-len(fixed(%c%1_1,0)));%c%1_2;
lprint repeat(" ",4-len(fixed(%c%1_2,0)));%c%1_3;
lprint repeat(" ",4-len(fixed(%c%1_3,0)));%c%1_4;
lprint repeat(" ",4-len(fixed(%c%1_4,0)));%c%1_1+%c%1_2+%c%1_3+%c%1_4
value1=value1+1
endwhile
```

COMMENT FORM FEED

```
lprint chr(12)
```

COMMENT CLEAR AWAY THE COUNTERS

```
call clear
```

COMMENT BACK TO STUDENT FILE

```
goto file %0students screen student1
return
```

COMMENT *****SQUADSUM*****

COMMENT PRODUCE SQUADRON SUMMARY REPORT

COMMENT ON INPUT TEXT1= SQUADRON NUMBER

COMMENT *****

procedure squadsum

```
%3=text1
```

```
menu print 20 58 4 15 Selecting Squadron %3
```

COMMENT ISOLATE THAT SQUADRON

```
query predefined squadron index temp2
```

```
order index temp2
```

COMMENT SORT BY SEMINAR

```
menu print 20 61 4 15 Sorting by Seminar
```

```
sort predefined semalf%9 index temp
```

COMMENT ZERO ALL THE COUNTERS-NEED TO KNOW WHICH SEMINARS ARE IN THIS

COMMENT SQUADRON. ENVIRONMENT FILE HAS THAT

```
goto file environ screen environ1
```

```
$frstunit=[fstsem%3]
```

```
$lastunit=[lstsem%3]
```

**** Project File SMSS ****

```

menu print 20 61 4 15 Preparing Counters
call czero
goto file %0students screen student1
order index temp
menu print 20 45 4 15 Computing squadron stats
goto record rec-number 1
COMMENT RUN THRU FILE-UPDATE COUNTS OF ALL ATTRIBUTES. THERE WILL BE 4
COMMENT DIFFERENT SETS OF COUNTERS FOR 4 SQUADRONS
label nextrec2
%1=val([mix%9])
call count
if record < records
    goto record next
    jump nextrec2
endif
COMMENT PRINT OUT THE REPORT WITH ALL ATTRIBUTE COUNTERS
menu print 20 28 4 15 Printing Squadron %3 Stats- Make sure printer is on
lprint
lprint repeat(" ",27);"SQUADRON ";"%3";" MIX STATISTICS"
lprint repeat(" ",38);"MIX ";"%9";repeat(" ",25);date1(today)
lprint
lprint "    ATTRIBUTE";repeat(" ",18);
COMMENT PRINT ALL THE SEMINAR NUMBERS ACROSS TOP LINE
value1=$firstunit
while value1<= $lastunit
    lprint repeat("0",2-len(fixed(value1,0)));value1;" ";
    value1=value1+1
endwhile
lprint " TOTAL"
lprint
value1=1
COMMENT THERE ARE 23 DIFFERENT ATTRIBUTES
COMMENT AND THE FIELD DESCRIPTIONS ARE IN THE ENVIRONMENT FILE
goto file environ screen environ1
menu print 20 52 4 15 Printing Squadron %3 Stats
while value1<=23
    %1=value1
COMMENT PRINT ATTRIBUTE NUMBER, LABEL ,COUNT FOR EACH SEMINAR, & TOTAL COUNT
lprint repeat(" ",2-len("%1"));"%1. ";[d%1];repeat(" ",27-len([d%1]));
value2=$firstunit
$tot=0
while value2<=$lastunit
    %2=value2
    lprint %c%1_%2;repeat(" ",3-len(fixed(%c%1_%2,0)));
    value2=value2+1
    $tot=$tot+%c%1_%2
endwhile
lprint $tot

```

**** Project File SMSS ****

```

value1=value1+1
endwhile
COMMENT FORM FEED
  lprint chr(12)
COMMENT CLEAR AWAY THE COUNTERS
  call clear
COMMENT BACK TO STUDENT FILE
  goto file %0students screen student1
return

```

```

COMMENT *****SEMSUM*****
COMMENT PRODUCE SEMINAR SUMMARY REPORT
COMMENT *****
procedure semsum
  menu print 20 58 4 15 Sorting by seminar
COMMENT SORT BY SEMINAR,RANK (KIND OF)-1ST IS MOST POTENTIAL SL, LAST IS LEAST
  sort predefined slorder%9 index temp
  order index temp
COMMENT DETERMINE WHICH OUTPUT MEDIA
  call whichcut
  report print semstat%9 %4 %5
return

```

```

COMMENT *****COUNT*****
COMMENT UPDATE ALL THE ATTRIBUTE COUNTS BASED ON THIS RECORD
COMMENT *****
procedure count
COMMENT IGNORE DELETED RECORDS
if deleted then return
COMMENT EACH "%C" COUNTER IS A COUNT OF THAT ATTRIBUTE
COMMENT %1 INDICATES WHICH UNIT (SQUADRON OR SEMINAR)
$c1_%1=$c1_%1+if [usaf]==="y" then 1 else 0
$c2_%1=$c2_%1+if [nomast]==="y" then 1 else 0
$c3_%1=$c3_%1+if [sorg]==="y" then 1 else 0
$c4_%1=$c4_%1+if [pilot]==="y" then 1 else 0
$c5_%1=$c5_%1+if [navigator]==="y" then 1 else 0
$c6_%1=$c6_%1+if [single]==="y" then 1 else 0
$c7_%1=$c7_%1+if [usafa]==="y" then 1 else 0
$c8_%1=$c8_%1+if [army]==="y" then 1 else 0
$c9_%1=$c9_%1+if [othercompl]==="y" then 1 else 0
$c10_%1=$c10_%1+if [minority]==="y" then 1 else 0
$c11_%1=$c11_%1+if [female]==="y" then 1 else 0
$c12_%1=$c12_%1+if [captain]==="y" then 1 else 0
$c13_%1=$c13_%1+if [nonline]==="y" then 1 else 0
$c14_%1=$c14_%1+if [stratopsk]==="y" then 1 else 0
$c15_%1=$c15_%1+if [tacopsk]==="y" then 1 else 0

```

**** Project File SMSS ****

```
$c16_1=$c16_1+if [ppbsk]==="y" then 1 else 0
$c17_1=$c17_1+if [acqlogsk]==="y" then 1 else 0
$c18_1=$c18_1+if [spacesk]==="y" then 1 else 0
$c19_1=$c19_1+if [tbd1]==="y" then 1 else 0
$c20_1=$c20_1+if [tbd2]==="y" then 1 else 0
$c21_1=$c21_1+if [tbd3]==="y" then 1 else 0
$c22_1=$c22_1+if [tbd4]==="y" then 1 else 0
$c23_1=$c23_1+if [tbd5]==="y" then 1 else 0
return
```

```
COMMENT *****CZERO*****
COMMENT ZERO ALL THE COUNTERS
COMMENT COUNTERS ARE $Cij WHERE i RANGES FROM 1 TO 23 AND
COMMENT j CAN BE 1 TO 4 FOR SQUADRONS IN THE SCHOOL SUMMARY
COMMENT OR ANY SEMINAR NUMBER IN THE SQUADRON SUMMARY
COMMENT ON INPUT, $FRSTUNIT & $LASTUNIT ARE 1 AND 4 FOR SCHOOL SUMMARY
COMMENT OR THE HIGHEST & LOWEST SEMINAR NUMBER FOR SQUADRON SUMMARY
COMMENT *****
```

```
procedure czero
value1=$frstunit
while value1 <= $lastunit
%1=value1
value2=1
COMMENT 23 COUNTERS FOR EACH UNIT
while value2<=23
%2=value2
$c%2_1=0
value2=value2+1
endwhile
value1=value1+1
endwhile
return
```

```
COMMENT *****CLEAR*****
COMMENT CLEAR ALL THE COUNTERS FROM MEMORY
COMMENT *****
```

```
procedure clear
value1=$frstunit
while value1 <= $lastunit
%1=value1
value2=1
COMMENT 23 COUNTERS FOR EACH UNIT
while value2<=23
%2=value2
clear $c%2_1
value2=value2+1
endwhile
```

**** Project File SMSS ****

```

value1=value1+1
endwhile
return

COMMENT *****GETSEM*****
COMMENT RUN APPROPRIATE QUERY BASED ON TEXT1 INPUT
COMMENT RETURN WITH TEMP INDEX FILE SET
COMMENT *****
procedure getsem
if text1=="all"
COMMENT SORT WHOLE FILE ON SEMINAR ORDER. REMEMBER %9 IS BASED ON $MIX
elseif left(text1,2)=="sq"
COMMENT SELECTING ONE WHOLE SQUADRON
    text1=mid(text1,3,1)
    menu print 20 59 4 15 Selecting Seminar(s)
    query predefined squadron index temp2
    order index temp2
COMMENT SORT BY SEMINAR/ALPHA
else
COMMENT SINGLE SEMINAR
COMMENT MAKE SURE IT'S IN RANGE
    if val(text1)<1 or val(text1) > 50
        text1="0"
        return
    endif
COMMENT IF HE ENTERED SINGLE DIGIT, FIX IT
    if len(text1)=1 then text1="0";text1
    menu print 20 56 4 15 Selecting Seminar
    query predefined mix%9 index temp2
    order index temp2
endif
    menu print 20 61 4 15 Sorting by seminar
    sort predefined semalf%9 index temp
return

COMMENT *****WHICHOUT*****
COMMENT USER SELECTS PRINT OUTPUT OPTION: PRINTER, DISK, OR SCREEN
COMMENT ON RETURN, %4= "PRINTER", "DISK", OR "SCREEN"
COMMENT %5= NULL, FILENAME OF DISK FILE, OR NULL
COMMENT *****

procedure whichout
label main1311
$lv|=1311
COMMENT DISPLAY SELECTION SCREEN
menu clear 15 4
menu draw box 2 31 5 45 15 4
menu print 3 32 15 4 REPORT OUTPUT

```

**** Project File SMSS ****

```
menu print 4 34 15 4 SELECTION
menu print 6 19 15 4 Choose where you want report output sent
menu print 8 36 15 4 OPTIONS
menu print 10 24 15 4 1. PRINTER
menu print 11 24 15 4 2. DISK
menu print 12 24 15 4 3. SCREEN
menu print 14 24 15 4 Default (Esc) is PRINTER
$firstline=10
$lastline=12
$linediff=1
$colm=23
call selopt
if $option=0 and $key=27 then $option=1
if $option=0 and ($key=104 or $key=72)
    call help
    jump main1311
endif
%6=null
COMMENT CALCULATE %4 AND %5
if $option=1
    %4="printer"
    %5=null
elseif $option=2
    %4="disk"
COMMENT GET DISKFILE NAME
    menu print 10 37 15 4 Enter file (or path) name--No extension
    beep
    menu input 11 37 4 15 20 text1
    if text1=null then jump main1311
    %5=text1
    %6="file, " ;"%5"; ".prt"
else
    %4="screen"
    %5=null
endif
menu print 20 30 4 15 Sending report to %4 %6
return
```

**** Project File ENVWRITE ****

```
COMMENT WRITE THE ASCII ENVIRONMENT RECORD
COMMENT FIRST ERASE ENV.ASC IF IT ALREADY EXISTS
goto file environ screen environ1
if file("env.asc")=1.0 then file erase env.asc
COMMENT REBUILD ONE-RECORD ASCII ENVIRONMENT FILE
COMMENT THIS ORDER SEEMS A LITTLE COMPLICATED BUT IT'S THE ORDER THAT MIXER
COMMENT NEEDS THEM IN. IF THE STRUCTURE OF ENVIRON EVER CHANGES, THESE FIELD
COMMENT NUMBERS WILL PROBABLY CHANGE, BUT THE RELATIVE ORDER MUST STAY THE
COMMENT SAME. THAT ORDER IS MIX,MEDIA,NUM.BUDS,FSTSEM1,LSTSEM1,MISSEM11,
COMMENT MISSEM12,MISSEM13,(SEMINAR DATA FOR SQDN 2,3,&
4),SEMCHANGE,BUDS,SPACESK,PPBSK,TACOPSK
COMMENT STRATOPSK,PILOT, ANV, SINGLE USAFA, ARMY, OTHRCOMP,MINORITY, FEMALE,
COMMENT CAPTAIN, NONLINE, SORG,SOSRULE,NOMAST,USAF,IORULE,TBD1,
COMMENT TBD2,TBD3,TBD4,TBD5. NOTE ARIRULE NOT USED.
write all [1;74;2;22;27;26;45;43;42;41;44;31;40;30;24;29;28;25;46;50] ascii
env.asc
```

**** Project File GOMIX ****

COMMENT PROJECT FILE GOMIX

COMMENT WRAPUP SMART SESSION AND EXIT OFF TO DOS TO DO MIXING

menu clear 15 4

menu print 20 51 4 15 Exit Smart. Go off to mixer

quit

**** Project File IODEL ****

```
COMMENT PROJECT IODEL- DELETE ALL THE INTERNATIONAL OFFICERS
COMMENT THEY'RE GONE AFTER 2ND MIX. THIS PROCESS MARKS THEM DELETED BUT DOES
COMMENT NOT PURGE THEIR RECORDS FROM THE STUDENT FILE
menu clear 15 4
menu print 3 24 15 4 SMSS - DELETE IO'S
menu draw box 2 23 4 42 15 4
menu print 5 2 15 4 This process should be done prior to mixing for 3rd mix
menu print 6 2 15 4 since the IO's will be gone then.
menu print 8 2 15 4 ARE YOU SURE YOU WANT TO DELETE IO'S(y/n)?
label iodel1
beep 1
menu input 8 45 4 15 1 $answer
if $answer=="n" then end
if $answer<>"y" and $answer<>"Y" then jump iodel1
COMMENT USE PREDEFINED QUERY TO DELETE IOS
menu print 20 40 4 15 Deleting International Officer Records
query predefined iodel neither
end
```

**** Project File LOADEM ****

```
COMMENT *****LOADEM*****
COMMENT INITIAL LOAD OF STUDENT DATA BASE & HELP SCREEN FILE
COMMENT %0 MUST BE SET OUT OF GETMEDIA OR NEWMIX
COMMENT *****
menu print 20 58 4 15 Loading Student file
load %0students screen student1
if cerror=3001
    beep 3
    message Error opening students database--give up
COMMENT TELL SMSS.BAT NOT TO DO MIXING
    unload all
    if file("env.asc")=1.0 then file erase env.asc
    quit
endif
activate %0students screen student2
menu print 20 54 4 15 Loading Help Screen file
activate help screen standard
```

**** Project File NEWMIX ****

```

COMMENT *****
COMMENT PROJECT NEWMIX-UPDATE STUDENT DB WITH RESULTS OF MIXING
COMMENT $media and %0 set on return IAW environment file
COMMENT *****

COMMENT FIRST GET MEDIA CODE OUT OF ENVIRONMENT FILE TO DETERMINE HOW WE'RE
COMMENT OPERATING. THIS WAS SET BY USER AT INITIAL ENTRY
goto file environ screen environ1
goto record rec-number 1
let $media=[media]
COMMENT FLOPPY?
if $media=1
    %0="b:\\"
COMMENT HARD DISK?
elseif $media=2
    %0=null
COMMENT FLOPPY/VDISK?
elseif $media=3
    %0="d:\\"
COMMENT ANYTHING ELSE MEANS A PROBLEM
else
    message ERROR in environment file media code
    stop
endif
COMMENT READ STUOUT.ASC (MIXER OUTPUT) INTO NEWMIX DATA BASE
COMMENT NEWMIX WILL BE THE TRANSACTION FILE TO UPDATE FIELDS IN STUDENTS
load %0newmix screen standard
menu print 20 37 4 15 Reading mixer interface file (stuout.asc)
COMMENT MIXER PASSES NAME,STU.NO, SQUADRON,MIXX,RULE PRIORITY,SLX,ASLX,CC,SRO
COMMENT WE'LL IGNORE NAME,CC,AND SRO
read ascii stuout.asc fields [0;stu.no;squadron;mixx;priority;slx;aslx]
COMMENT PREPARE FOR TRANSACTION UPDATE. LOAD STUDENT FILE
execute loadem in-memory
order key [stu.no]
split horizontal 11 2
goto window 2
goto file %0newmix screen standard
menu print 19 33 4 15 ***UPDATING STUDENT FILE WITH NEW MIX DATA***
COMMENT DETERMINE TIME TO RUN
if $media=1
    %1="33"
elseif $media=2
    %1="9"
elseif $media=3
    %1="15"
endif
menu print 20 33 15 4 This will take about %1 minutes. Please wait.
transactions predefined newmix no-audit

```

**** Project File NEWMIX ****

COMMENT PREDEFINED TRANSACTION SET UP TO DELETE RECORDS-NOW PURGE THEM
close
unload file %0newmix
menu print 20 46 4 15 Purging used transaction records
utilities purge %0newmix
COMMENT DELETE FILE STUOUT.ASC TO SHOW UPDATE HAS BEEN DONE
repaint
file erase stuout.asc

**** Project File RAMLOAD ****

```
COMMENT *****RAMLOAD*****
COMMENT LOAD STUDENT (NEWMIX) DATA BASES INTO VIRTUAL MEMORY STORAGE
COMMENT VIRTUAL MEMORY OPERATES MUCH FASTER THAN FLOPPY DISK
COMMENT SYSTEM MUST HAVE BEEN BOOTED WITH THE FOLLOWING COMMAND IN THE ROOT
COMMENT CONFIG.SYS FILE---DEVICE=\BIN\VDISK.SYS 220
COMMENT *****
menu print 16 2 4 15 NOTE:
menu print 16 7 15 4 This option requires that VDISK be setup in the root
config.sys file
menu print 17 7 15 4 at startup time. If not, go back to DOS & do it.
menu print 18 7 15 4 Is VDISK configured? (y/n)
label rloop
beep
menu input 18 34 4 15 1 text1
COMMENT IF USER BAILS OUT RETURN WITH TEXT1=N
if text1=="n" then end
if text1<>"y" and text1<>"Y" then jump rloop
COMMENT COPY ALL STUDENTS & NEWMIX FILES TO VDISK
menu print 20 47 4 15 Loading Student file to ramdisk
file copy b:\students.db to d:students.db
file copy b:\students.dbs to d:students.dbs
COMMENT INDEX FILES-IF STRUCTURE OF STUDENTS CHANGES, THESE MAY HAVE TO CHANGE
file copy b:\students.i01 to d:students.i01
file copy b:\students.i06 to d:students.i06
file copy b:\students.i21 to d:students.i21
file copy b:\newmix.db to d:newmix.db
file copy b:\newmix.dbs to d:newmix.dbs
```

**** Project File RAMSAVE ****

```
COMMENT *****RAMSAVE*****
COMMENT SAVE VDISK FILES BACK TO FLOPPY AFTER SMSS SESSION ENDS
COMMENT *****
menu print 20 46 4 15 Reclaiming vdisk files to floppy
COMMENT FILE COPY DOESN'T OVERWRITE-SO FIRST DELETE DESTINATION FILES
file erase b:students.*
file erase b:newmix.db*
COMMENT VOLATILE VDISK BACK TO PERMANENT FLOPPY
file copy d:students.db to b:students.db
file copy d:students.dbs to b:students.dbs
COMMENT INDEX FILES-NOTE IF FILE STRUCTURE CHANGES-THESE MAY HAVE TO CHANGE
file copy d:students.i01 to b:students.i01
file copy d:students.i06 to b:students.i06
file copy d:students.i21 to b:students.i21
file copy d:newmix.db to b:newmix.db
file copy d:newmix.dbs to b:newmix.dbs
COMMENT ERASE RAMDISK FILES SO DON'T GET "FILE EXISTS" MESSAGE NEXT TIME
file erase d:*. *
repaint
```

**** Project File STUNUMB ****

```
COMMENT PROJECT STUNUMB-CREATE STUDENT NUMBERS
COMMENT THIS SHOULD BE A ONE TIME OPERATION DONE WHEN STUDENT FILE
COMMENT IS SETTLED
COMMENT SET UP SCREEN-ISSUE WARNING MESSAGE
menu clear 15 4
menu print 3 24 15 4 SMSS - CREATE STUDENT NUMBERS
menu draw box 2 23 4 53 15 4
menu print 5 2 15 4 This process should generally be done once at the start of
menu print 6 2 15 4 the school year after all the records have been added to
menu print 7 2 15 4 the student file. Once you've assigned student numbers,
menu print 8 2 15 4 you shouldn't do this anymore.
menu print 10 2 15 4 ARE YOU SURE YOU WANT TO ASSIGN NEW STUDENT NUMBERS (y/n)?
label stun1
beep 1
menu input 10 61 4 15 1 $answer
if $answer=="n" then end
if $answer<>"y" and $answer<>"Y" then jump stun1
COMMENT PUT FILE IN ALPHA ORDER
order key [name]
COMMENT USE PREDEFINED QUERY TO UPDATE EACH [STU.NO] TO 10,15,20,...ETC
menu print 20 31 4 15 Assigning student number to each student record
query predefined stunum neither
COMMENT REORGANIZE KEY FOR NEW VALUES
menu print 20 44 4 15 Building student number key file
key update
end
```

**** Project File STUREAD ****

```
COMMENT PROJECT STUREAD-READ IN THE STUDENT DATA BASE FROM ASCII DISK
COMMENT ON INPUT, $option IS SET TO 1 OR 2 FROM LEVEL 111 OPTION SCREEN
call header
COMMENT READ IN ASCII FILE- FIRST GET FILENAME
menu print 7 2 15 4 ENTER NAME OF INPUT FILE (e.g. STUDENT.ASC)
menu print 8 2 15 4 (Esc to exit).
beep
menu input 7 46 4 15 12 text1
COMMENT LET HIM BAIL OUT HERE TOO
if text1=null then end
COMMENT OPTION 1 MEANS BUILD FROM SCRATCH-DELETE ALL RECORDS
if $option = 1 then call deletem
call header
%1=text1
message Load source diskette in drive A:--Any key when ready
menu print 20 41 4 15 Reading source file into Student file
COMMENT FIELD LIST MAY HAVE TO CHANGE DEPENDING ON AU/DP
COMMENT NEXT IS INTERIM FORMAT FROM BUCKNER'S DATABASE
READ ASCII A:\%1 FIELDS [1;6;2;9;8;12;19;26;3;5;7;11;18;10;20;22;27;23;25;13;17;
28;31;33]
COMMENT UPDATE ALL THE KEYS
menu print 20 36 4 15 Organizing all key indexes
key update
COMMENT IF FLOPPY , MAKE SURE WE GET RUNTIME DISK BACK IN A:
if $media=1 or $media=3 then message Load SMSS execution disk back in A:--Any
key when ready
end

COMMENT *****HEADER*****
COMMENT PAINTS THE SCREEN PROPERLY
COMMENT *****
procedure header
menu clear 15 4
menu print 3 26 15 4 SMSS - BUILD STUDENT FILE
menu draw box 2 25 4 51 15 4
return

COMMENT *****DELETEM*****
COMMENT CLEAR OUT EXISTING STUDENT DATA BASE BY DELETING ALL RECORDS
COMMENT *****
procedure deletem
menu print 5 2 15 4 YOU ARE ABOUT TO ERASE ALL CURRENT DATA BASE RECORDS
menu print 6 2 15 4 ARE YOU SURE YOU WANT TO DO THIS (y/n)?
label dell
beep 3
menu input 6 41 4 15 1 $answer
if $answer == "n" then end
if $answer <>"y" and $answer <>"Y" then jump dell
```

AD-A192 422

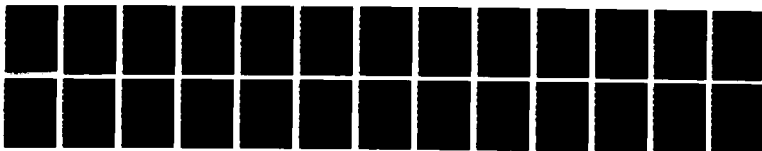
STUDENT MIX SOFTWARE SYSTEM (SMSS) REHOST(U) AIR
COMMAND AND STAFF COLL MAXWELL AFB AL
A G DECELLES ET AL APR 88 ACSC-88-0700

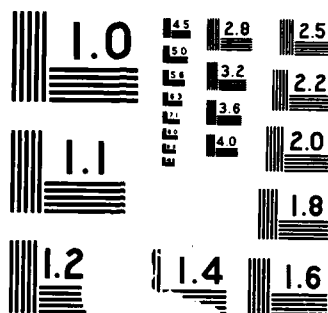
2/2

UNCLASSIFIED

F/G 12/5

NL





**** Project File STUREAD ****

COMMENT THAT'S ENOUGH WARNING-HE REALLY WANTS TO DO IT
COMMENT PREDEFINED QUERY "DELALL" DELETES ALL ACTIVE RECORDS-USE IT
menu print 20 33 4 15 Deleting all old records in Student file
query predefined delall neither
unload file %0students
menu print 20 33 4 15 Purging all deleted records from Student file
utilities purge %0students
load %0students screen student1
return

**** Project File STUWRITE ****

```
COMMENT *****
COMMENT PROJECT STUWRITE
COMMENT WRITE THE ASCII STUDENT FILE TO PASS TO THE MIXER
COMMENT *****
goto file %0students screen student1
COMMENT ORDER BY RANK, DOR, TAFCSO
menu print 20 64 4 15 Sorting by rank
sort predefined dor index temp
order index temp
COMMENT GET RID OF LAST VERSION OF ASCII FILE IF IT EXISTS
if file ("stu.asc")=1.0 then file erase stu.asc
COMMENT BUILD THE ASCII FILE FOR MIXING
menu print 20 40 4 15 Writing mixer interface file (stu.asc)
write all [1;28;31;33;39;42;65] ascii stu.asc
```

Appendix Three

SMSS Mixer Program Program Listing

The mixer is a program, written in GW-BASIC, that does student mixing according to the rules passed in the environment record.

```

10000 REM *****
10010 REM **
10020 REM ** PROGRAM NAME: STUDENT MIX SOFTWARE SYSTEM (SMSS)
10030 REM **
10040 REM ** FUNCTION: MIX ACSC STUDENTS
10050 REM ** FILE NAME: MIXER.BAS DATE: 16 FEB 1988
10060 REM ** COMPUTER: ZENITH 158 LANGUAGE: BASIC
10070 REM ** AUTHOR: ART DECELLES
10080 REM **
10090 REM *****
10100 REM **
10110 REM ** DESCRIPTION:
10120 REM **
10130 REM ** THIS PROGRAM OPTIMALLY MIXES ACSC STUDENTS BASED ON STUDENT
10140 REM ** CHARACTERISTICS AND RULE PRIORITIES [NINE (HIGHEST) TO ZERO
10150 REM ** (LOWEST)]. MIXING OCCURS IN THE FOLLOWING GENERAL SEQUENCE:
10160 REM **
10170 REM ** 1. SEMINAR LEADERS AND THEIR ASSISTANTS (SL/ASL) ARE
10180 REM ** ASSIGNED IN THE FIRST PASS OF THE STUDENT DATA. ALSO,
10190 REM ** PRE-ASSIGNED STUDENTS ARE "LOCKED-IN". INTERNATIONAL
10200 REM ** OFFICERS AND SOS STUDENTS DON'T CHANGE SEMINARS IF THEY
10210 REM ** ARE ASSIGNED A PRIORITY OF "9".
10220 REM **
10230 REM ** 2. ALL OTHER STUDENTS ARE RANDOMLY ASSIGNED TO SEMINARS
10240 REM ** BASED ON THEIR CHARACTERISTICS, EACH CONSIDERED IN ORDER
10250 REM ** OF RULE PRIORITY ESTABLISHED BY THE MIX MASTER.
10260 REM **
10270 REM ** A. STUDENTS CANNOT BE ASSIGNED TO A PREVIOUS SEMINAR.
10280 REM ** B. ASSIGNMENT IS MADE IF ALLOCATIONS PERMIT.
10290 REM ** C. IF CONFLICT OCCURS MIXER ATTEMPTS TO MAKE A ONE FOR
10300 REM ** ONE EXCHANGE WITHIN ALLOCATION CONSTRAINTS.
10310 REM ** D. IF STILL NOT POSSIBLE A FORCED ASSIGNMENT IS MADE:
10320 REM **
10330 REM ** a. IF AN ALLOCATED SLOT IS AVAILABLE FOR THE RULE
10340 REM ** BEING PROCESSED.
10350 REM ** b. IF NOT, A ONE FOR ONE EXCHANGE IS ATTEMPTED WITH
10360 REM ** ASSIGNMENTS OF EQUAL PRIORITY.
10370 REM ** C. AS A LAST RESORT, ONE FOR ONE EXCHANGES ARE
10380 REM ** ATTEMPTED FOR INCREMENTALLY HIGHER PRIORITIES.
10390 REM **
10400 REM *****
10410 REM **
10420 REM ** MAJOR VARIABLES:
10430 REM **
10440 REM ** ALOT(27,4,11): ARRAY CONTAINING ALLOCATIONS WHICH REPRESENT
10450 REM ** THE TARGET BALANCED CHARACTERISTICS.
10460 REM **
10470 REM ** STAT(27,4,11): ARRAY TO GATHER STATISTICS ON ASSIGNMENTS.
10480 REM **

```

```

10490 REM ** ASSIGN(44,13,5): ARRAY CONTAINING A SUBSET OF STUDENT INFO **
10500 REM ** AFTER AN ASSIGNMENT IS MADE. **
10510 REM ** **
10520 REM ** CHAR(27): ARRAY WHICH CONTAINS THE CHARACTERISTICS FOR THE **
10530 REM ** STUDENT CURRENTLY BEING CONSIDERED. **
10540 REM ** **
10550 REM ** SEMI(12): ARRAY MAINTAINING CONSTRAINT INFORMATION FOR **
10560 REM ** ATTEMPTS TO ASSIGN STUDENT TO SEMINARS. **
10570 REM ** **
10580 REM ** RECNUM(575): ARRAY INDICATING WHICH RECORDS HAVE/HAVEN'T **
10590 REM ** BEEN ASSIGNED PREVIOUSLY. **
10600 REM ** **
10610 REM ** MS(3): ARRAY INDICATING MISSING SEMINARS IN A SQUADRON. **
10620 REM ** **
10630 REM ** SQ$(4,7): ARRAY CONTAINING SCHOOL/SQUADRON PARAMETERS. **
10640 REM ** **
10650 REM ** SL(4): ARRAY USED TO TRACK SL ASSIGNMENTS. **
10660 REM ** **
10670 REM ** ASL(4): ARRAY USED TO TRACK ASL ASSIGNMENTS. **
10680 REM ** **
10690 REM ** STAT$(27): ARRAY CONTAINING RULE ALPHA DESCRIPTORS. **
10700 REM ** **
10710 REM ** RULE$(27): ARRAY CONTAINING RULE PRIORITIES. **
10720 REM ** **
10730 REM ***** **
10740 REM ** **
10750 REM ** STORAGE REQUIREMENTS: **
10760 REM ** **
10770 REM **
10780 DIM SQ$(4,7), STAT(27,4,11), RULE$(27), ASSIGN(44,13,5), MS(3)
10790 DIM TEMPIN$(36), STAT$(27), ALOT(27,4,11), SL(4), ASL(4)
10800 DIM CHAR(27), SEMI(12), RECNUM(575)
10810 REM
10820 REM ** **
10830 REM ***** **
10840 REM ** **
10850 REM ** INPUT/OUTPUT: **
10860 REM ** **
10870 REM **
10880 REM FIELD #2, 27 AS SNAME$, 4 AS STN$, 1 AS ASQ$, 2 AS MIX1$, **
10890 REM 2 AS MIX2$, 2 AS MIX3$, 2 AS MIXX$, 1 AS USAF$, **
10900 REM 1 AS NOED$, 1 AS HORG$, 1 AS PILOT$, 1 AS NAV$, **
10910 REM 1 AS SING$, 1 AS AFA$, 1 AS ARMY$, 1 AS REST$, **
10920 REM 1 AS RACE$, 1 AS SLX$, 1 AS ASLX$, 1 AS PRIOR$ **
10930 REM **
10940 REM FIELD #3, 1 AS TOPS$, 1 AS SOPS, 1 AS PPBS$, 1 AS ACQLOG$, **
10950 REM 1 AS SPAACE$, 1 AS TBD1$, 1 AS TBD2$, 1 AS TBD3$, **
10960 REM 1 AS TBD4$, 1 AS TBD5$, 1 AS SL$, 1 AS ASL$, 1 AS SOS$, **
10970 REM 1 AS ARI$, 1 AS SRO$, 1 AS CC$, 1 AS IO$, 1 AS FEM$, **

```

```

10980 REM          1 AS RNK$, 1 AS NOLINE$                **
10990 REM                                                    **
11000 REM *****
11010 REM **                                                    **
11020 REM **    PROCEDURE:                                    **
11030 REM **                                                    **
11040 REM
11050 SCREEN 0,1
11060 COLOR 2 0,1
11070 OPEN "0", #1, "ERROR.SMS": PRINT #1, "SMSS MIXER PROGRAM ERROR":CLOSE
11080 DELTA = +1
11090 GOSUB 13070                      :REM ** RETRIEVE SCHOOL DATA **
11100 CLS: LN = 3
11110 LOCATE 2,18: PRINT "TURN THE TRACE OUTPUT OFF AT TERMINAL? (Y/N) ";
11120 INPUT "", CH$: LOCATE LN,18
11130 IF CH$ = "Y" OR CH$ = "y" THEN TRONS = 0: TRONS$ = "OFF": GOTO 11200
11140 IF CH$ = "N" OR CH$ = "n" THEN TRONS = 1: TRONS$ = "ON": GOTO 11200
11150 IF CH$ = "D" OR CH$ = "d" THEN TRONS = 2: TRONS$ = "ON": GOTO 11200
11160 LN = LN + 1
11170 IF LN = 25 THEN GOTO 11100
11180 PRINT "          INCORRECT CHOICE - TRY AGAIN? ";
11190 GOTO 11120
11200 CLS: LOCATE 2,32: PRINT "MIX ";MIX;" RULES";
11210 LOCATE 4,21: PRINT "PROCESSING MIXER INPUT WITH TRACE "; TRONS$:PRINT
11220 FOR I = 1 TO 27: INPUT #1, RULE$(I): NEXT I      :REM ** INIT RULES **
11230 CLOSE #1
11240 GOSUB 12850                      :REM ** INITIALIZE RULE DESCRIPTORS **
11250 GOSUB 13770                      :REM ** INITIALIZE STATISTICS **
11260 IOMODE$ = "I": GOSUB 12210      :REM ** PROCESS INPUT DATA **
11270 IOMODE$ = "O": GOSUB 13980      :REM ** INITIALIZE RANDOM # GENERATOR **
11280 OPEN "0", #1, "MIXERLOG.DAT"
11290 IF MIX = 1 THEN GOSUB 19900      :REM ** ALLOC SQ SLOTS **
11300 GOSUB 14100                      :REM ** ALLOC SEM SLOTS **
11310 CLS: LOCATE 2,30: PRINT "SMSS MIXING ROUTINE": PRINT
11320 PRIOR = 9
11330 GOSUB 16230                      :REM ** ASGN SL/ASL **
11340 FOR PRIOR = 9 TO 1 STEP -1
11350     FOR INDEX = 3 TO 27
11360         IF VAL(RULE$(INDEX)) <> PRIOR THEN GOTO 11410
11370         IF PRIOR = 9 AND INDEX = 22 THEN GOTO 11410
11380         IF PRIOR = 9 AND INDEX = 19 THEN GOTO 11410
11390         IR = INDEX: RL = INDEX: RL$ = STAT$(IR)
11400         GOSUB 18700
11410     NEXT INDEX
11420 NEXT PRIOR
11430 IR = 0: INDEX = 0: RL = 0: PRIOR = 0: RL$ = "ALL OTHER": STAT$(0)=RL$
11440 GOSUB 18700                      :REM ** PROCESS REST **
11450 CLOSE #1: GOSUB 20170            :REM ** OUTPUT RSLTS **
11460 OPEN "I", #1, "MIXERLOG.DAT"    :REM ** FINISHED **

```

```

11470     FOR I = 1 TO 999
11480         IF EOF(1) GOTO 11510
11490         INPUT #1, TMP$: LPRINT TMP$
11500     NEXT I
11510     CLOSE: GOSUB 11610                                :REM ** OUTPUT STATS **
11520     CLOSE: KILL "ERROR.SMS"
11530     END
11540 REM
11550 REM **
11560 REM *****
11570 REM **
11580 REM **     SUBROUTINE TO OUTPUT MIXER STATISTICS
11590 REM **
11600 REM
11610     LPRINT: LPRINT: LPRINT: LPRINT
11620     STAT$(0) = "SEMINAR ASSIGNMENTS"
11630     FOR I = 0 TO 27
11640         IF I = 1 OR I = 2 THEN GOTO 11710
11650         LPRINT "* STATISTICS ON "; STAT$(I); "     PRI = "; RULE$(I)
11660         FOR J = 1 TO 4
11670             LPRINT "SQ: "; J; " => ";
11680             FOR K = 1 TO 11: LPRINT STAT(I,J,K);: NEXT K
11690             LPRINT " "
11700         NEXT J: LPRINT " "
11710     NEXT I
11720     LPRINT "* ASSIGNMENT ARRAY: "
11730     FOR I = 1 TO 44
11740         IF I > 11 THEN FS = 12 ELSE FS = 1
11750         IF I > 22 THEN FS = 23
11760         IF I > 33 THEN FS = 34
11770         LPRINT "SEM: "; I; " MIX1 STNTS=";
11780         FOR J = 1 TO 11: CHAR(J) = 0: NEXT J
11790         FOR J = 1 TO 13
11800             IF ASSIGN(I,J,1) = 0 GOTO 11820
11810             CHAR(ASSIGN(I,J,1)-FS+1) = CHAR(ASSIGN(I,J,1)-FS+1) + 1
11820             LPRINT ASSIGN(I,J,1);
11830         NEXT J
11840         TEMP = 0
11850         FOR J = 1 TO 11
11860             IF TEMP < CHAR(J) THEN TEMP = CHAR(J)
11870         NEXT J
11880         LPRINT " MAX#"; TEMP
11890         LPRINT "SEM: "; I; " MIX2 STNTS=";
11900         FOR J = 1 TO 11: CHAR(J) = 0: NEXT J
11910         FOR J = 1 TO 13
11920             IF ASSIGN(I,J,2) = 0 GOTO 11940
11930             CHAR(ASSIGN(I,J,2)-FS+1) = CHAR(ASSIGN(I,J,2)-FS+1) + 1
11940             LPRINT ASSIGN(I,J,2);
11950         NEXT J

```

```

11960      TEMP = 0
11970      FOR J = 1 TO 11
11980          IF TEMP < CHAR(J) THEN TEMP = CHAR(J)
11990      NEXT J
12000      LPRINT " MAX#"; TEMP
12010      LPRINT "SEM:"; I; " RULE#=";
12020      FOR J = 1 TO 13
12030          LPRINT ASSIGN(I,J,5);
12040      NEXT J
12050      LPRINT " "
12060      LPRINT "SEM:"; I; " RECORD#=";
12070      FOR J = 1 TO 13
12080          LPRINT ASSIGN(I,J,0);
12090      NEXT J
12100      LPRINT " "
12110      LPRINT " "
12120  NEXT I
12130  RETURN
12140 REM
12150 REM **
12160 REM *****
12170 REM **
12180 REM ** SUBROUTINE TO PROCESS INPUT DATA
12190 REM **
12200 REM
12210 OPEN "I", #1, "STU.ASC"
12220 IF MEDIA$ <> "1" THEN GOTO 12260
12230 OPEN "R", #2, "B:STDNT1.DAT"
12240 OPEN "R", #3, "B:STDNT2.DAT"
12250 GOTO 12280
12260 OPEN "R", #2, "STDNT1.DAT"
12270 OPEN "R", #3, "STDNT2.DAT"
12280 FIELD #2, 27 AS SNAME$, 4 AS STN$, 1 AS ASQ$, 2 AS MIX1$, 2 AS MIX2$, 2 AS
MIX3$, 2 AS MIXX$, 1 AS USAF$, 1 AS NOED$, 1 AS HORG$, 1 AS PILOT$, 1 AS NAV$, 1 AS
SING$, 1 AS AFA$, 1 AS ARMY$, 1 AS REST$, 1 AS RACE$, 1 AS SLX$, 1 AS ASLX$, 1 AS
PRIOR$
12290 FIELD #3, 1 AS TOPS$, 1 AS SOPS$, 1 AS PPBS$, 1 AS ACQLOG$, 1 AS
SPAACE$, 1 AS TBD1$, 1 AS TBD2$, 1 AS TBD3$, 1 AS TBD4$, 1 AS TBD5$, 1 AS SL$, 1 AS
ASL$, 1 AS SOS$, 1 AS ARI$, 1 AS SRO$, 1 AS CC$, 1 AS IO$, 1 AS FEM$, 1 AS RNK$, 1 AS
NOLINE$
12300 SI = 0: KO = 1: PRINT: PRINT " READING ";
12310 FOR KS = 1 TO 575
12320     IF KS/10 >= KO THEN PRINT ".": KO = KO + 1
12330     STDNT = SI
12340     FOR J = 1 TO 36
12350         IF EOF (1) THEN GOTO 12680
12360         INPUT #1, TEMPIN$(J)
12370     NEXT J
12380     LSET SNAME$ = TEMPIN$(1): LSET IO$ = TEMPIN$(21)

```

```

12390      LSET ASQ$ = TEMPIN$(2):      LSET USAF$ = TEMPIN$(22)
12400      LSET MIX1$ = TEMPIN$(3):      LSET NOED$ = TEMPIN$(23)
12410      LSET MIX2$ = TEMPIN$(4):      LSET HORG$ = TEMPIN$(24)
12420      LSET MIX3$ = TEMPIN$(5):      LSET PILOT$ = TEMPIN$(25)
12430      LSET STN$ = TEMPIN$(6):      LSET NAV$ = TEMPIN$(26)
12440      LSET SOS$ = TEMPIN$(7):      LSET SING$ = TEMPIN$(27)
12450      LSET ARI$ = TEMPIN$(8):      LSET AFA$ = TEMPIN$(28)
12460      LSET CC$ = TEMPIN$(9):      LSET ARMY$ = TEMPIN$(29)
12470      LSET SRO$ = TEMPIN$(10):      LSET REST$ = TEMPIN$(30)
12480      LSET SL$ = TEMPIN$(11):      LSET RACE$ = TEMPIN$(31)
12490      LSET ASL$ = TEMPIN$(12):      LSET FEM$ = TEMPIN$(32)
12500      LSET PPBS$ = TEMPIN$(13):      LSET RNK$ = TEMPIN$(33)
12510      LSET ACQLOG$ = TEMPIN$(14):      LSET NOLINE$ = TEMPIN$(34)
12520      LSET SPAACE$ = TEMPIN$(15):      LSET SOPS$ = TEMPIN$(35)
12530      LSET TBD1$ = TEMPIN$(16):      LSET TOPS$ = TEMPIN$(36)
12540      LSET TBD2$ = TEMPIN$(17):      LSET SLX$ = "0"
12550      LSET TBD3$ = TEMPIN$(18):      LSET ASLX$ = "0"
12560      LSET TBD4$ = TEMPIN$(19):      LSET MIXX$ = "0"
12570      LSET TBD5$ = TEMPIN$(20):      LSET PRIOR$ = "0"
12580      IF (IO$="Y" OR IO$="y") AND RULE$(22)="0" AND MIX=3 GOTO 12670
12590      JJ = 0: KK = 0: GOSUB 13270      :REM ** RECORD SCHOOL STATS **
12600      IF ASQ$ = " " GOTO 12650
12610      FOR J = 1 TO 4      :REM ** POST SQUADRON STATS **
12620          IF VAL(ASQ$) <> VAL(SQ$(J,1)) GOTO 12640
12630          JJ = J: KK = 0: GOSUB 13270: GOTO 12650
12640      NEXT J
12650      SI = SI + 1
12660      PUT #2,SI: PUT #3,SI
12670  NEXT KS
12680  STDNT = STAT(0,0,0): CLS
12690  LOCATE 2,25: PRINT "PROCESSED";KS-1;"STUDENTS INTO MIXER"
12700  LOCATE 3,20: PRINT STDNT; "STUDENTS KEPT FOR FURTHER PROCESSING"
12710  PRINT
12720  FOR I = 3 TO 27
12730      IF STAT(I,0,0) = 0 THEN GOTO 12750
12740      PRINT "  THERE WERE "; STAT(I,0,0); " "; STAT$(I); " STUDENTS"
12750  NEXT I
12760  PRINT "  **** ALL OTHER CHARACTERISTICS WERE NOT REPRESENTED"
12770  CLOSE #1: RETURN
12780 REM
12790 REM **
12800 REM *****
12810 REM **
12820 REM ** SUBROUTINE TO INITIALIZE RULE DESCRIPTIONS
12830 REM **
12840 REM
12850      STAT$(1) = "SEMINAR CHANGE":      STAT$(2) = "BUDDY RULE"
12860      STAT$(3) = "SPACE SKILL":      STAT$(4) = "PPBS SKILL"
12870      STAT$(5) = "TAC OPS SKILL":      STAT$(6) = "STRAT OPS SKILL"

```

```

12880     STAT$(7) = "ACQ/LOG SKILL":          STAT$(8) = "PILOT"
12890     STAT$(9) = "NAVIGATOR":              STAT$(10) = "SINGLE/UNACCOMP"
12900     STAT$(11) = "USFA GRADUATE":         STAT$(12) = "ARMY"
12910     STAT$(13) = "RES/NG/USN/USMC/CIV":  STAT$(14) = "MINORITY"
12920     STAT$(15) = "FEMALE":               STAT$(16) = "CAPTAIN"
12930     STAT$(17) = "NON LINE":             STAT$(18) = "SR ORG EXPERIENCE"
12940     STAT$(19) = "SOS":                 STAT$(20) = "NON MASTER DEGREE"
12950     STAT$(21) = "USAF":                 STAT$(22) = "IO"
12960     STAT$(23) = "UNLABELED":           STAT$(24) = "UNLABELED"
12970     STAT$(25) = "UNLABELED":           STAT$(26) = "UNLABELED"
12980     STAT$(27) = "UNLABELED"
12990     RETURN
13000 REM
13010 REM **
13020 REM *****
13030 REM **
13040 REM **   SUBROUTINE TO RETRIEVE SCHOOL DATA
13050 REM **
13060 REM
13070     OPEN "I",#1,"ENV.ASC"
13080     INPUT #1, MIX$, MEDIA$, PREV$
13090     FOR I = 1 TO 4
13100         ADJ = 0
13110         INPUT #1,SQ$(I,2),SQ$(I,3),SQ$(I,4),SQ$(I,5),SQ$(I,6)
13120         SQ$(I,1) = STR$(I)
13130         IF VAL(SQ$(I,4)) <> 0 THEN ADJ = ADJ + 1
13140         IF VAL(SQ$(I,5)) <> 0 THEN ADJ = ADJ + 1
13150         IF VAL(SQ$(I,6)) <> 0 THEN ADJ = ADJ + 1
13160         SQ$(I,7) = STR$(VAL(SQ$(I,3)) - VAL(SQ$(I,2)) - ADJ + 1)
13170     NEXT I
13180     MIX = VAL(MIX$)
13190     RETURN
13200 REM
13210 REM **
13220 REM *****
13230 REM **
13240 REM **   SUBROUTINE TO UPDATE STATISTICS
13250 REM **
13260 REM
13270     IF IOMODE$ = "0" THEN GOTO 13290
13280     IF JJ = 0 THEN STAT(0,0,0) = STAT(0,0,0) + DELTA
13290     IF KK <> 0 THEN STAT(0,JJ,KK) = STAT(0,JJ,KK) + DELTA
13300     IF JJ <> 0 THEN STAT(0,JJ,0) = STAT(0,JJ,0) + DELTA
13310     GOSUB 13440
13320     FOR II = 3 TO 27
13330         IF CHAR(II) = 0 THEN GOTO 13350
13340         STAT(II,JJ,KK) = STAT(II,JJ,KK) + DELTA
13350     NEXT II
13360     RETURN

```

```

13370 REM
13380 REM **
13390 REM *****
13400 REM **
13410 REM ** SUBROUTINE TO INITIALIZE CHARACTERISTICS ARRAY
13420 REM **
13430 REM
13440 IF SPAACE$ = "Y" OR SPAACE$ = "y" THEN CHAR(3) = 1 ELSE CHAR(3) = 0
13450 IF PPBS$ = "Y" OR PPBS$ = "y" THEN CHAR(4) = 1 ELSE CHAR(4) = 0
13460 IF TOPS$ = "Y" OR TOPS$ = "y" THEN CHAR(5) = 1 ELSE CHAR(5) = 0
13470 IF SOPS$ = "Y" OR SOPS$ = "y" THEN CHAR(6) = 1 ELSE CHAR(6) = 0
13480 IF ACQLOG$ = "Y" OR ACQLOG$ = "y" THEN CHAR(7) = 1 ELSE CHAR(7) = 0
13490 IF PILOT$ = "Y" OR PILOT$ = "y" THEN CHAR(8) = 1 ELSE CHAR(8) = 0
13500 IF NAV$ = "Y" OR NAV$ = "y" THEN CHAR(9) = 1 ELSE CHAR(9) = 0
13510 IF SING$ = "Y" OR SING$ = "y" THEN CHAR(10) = 1 ELSE CHAR(10) = 0
13520 IF AFA$ = "Y" OR AFA$ = "y" THEN CHAR(11) = 1 ELSE CHAR(11) = 0
13530 IF ARMY$ = "Y" OR ARMY$ = "y" THEN CHAR(12) = 1 ELSE CHAR(12) = 0
13540 IF REST$ = "Y" OR REST$ = "y" THEN CHAR(13) = 1 ELSE CHAR(13) = 0
13550 IF RACE$ = "Y" OR RACE$ = "y" THEN CHAR(14) = 1 ELSE CHAR(14) = 0
13560 IF FEM$ = "Y" OR FEM$ = "y" THEN CHAR(15) = 1 ELSE CHAR(15) = 0
13570 IF RNK$ = "Y" OR RNK$ = "y" THEN CHAR(16) = 1 ELSE CHAR(16) = 0
13580 IF NOLINE$ = "Y" OR NOLINE$ = "y" THEN CHAR(17) = 1 ELSE CHAR(17) = 0
13590 IF HORG$ = "Y" OR HORG$ = "y" THEN CHAR(18) = 1 ELSE CHAR(18) = 0
13600 IF SOS$ = "Y" OR SOS$ = "y" THEN CHAR(19) = 1 ELSE CHAR(19) = 0
13610 IF NOED$ = "Y" OR NOED$ = "y" THEN CHAR(20) = 1 ELSE CHAR(20) = 0
13620 IF USAF$ = "Y" OR USAF$ = "y" THEN CHAR(21) = 1 ELSE CHAR(21) = 0
13630 IF IO$ = "Y" OR IO$ = "y" THEN CHAR(22) = 1 ELSE CHAR(22) = 0
13640 IF TBD1$ = "Y" OR TBD1$ = "y" THEN CHAR(23) = 1 ELSE CHAR(23) = 0
13650 IF TBD2$ = "Y" OR TBD2$ = "y" THEN CHAR(24) = 1 ELSE CHAR(24) = 0
13660 IF TBD3$ = "Y" OR TBD3$ = "y" THEN CHAR(25) = 1 ELSE CHAR(25) = 0
13670 IF TBD4$ = "Y" OR TBD4$ = "y" THEN CHAR(26) = 1 ELSE CHAR(26) = 0
13680 IF TBD5$ = "Y" OR TBD5$ = "y" THEN CHAR(27) = 1 ELSE CHAR(27) = 0
13690 RETURN
13700 REM
13710 REM **
13720 REM *****
13730 REM **
13740 REM ** SUBROUTINE TO INITIALIZE/OUTPUT STATS & SEMINAR ASSIGNMENTS
13750 REM **
13760 REM
13770 PRINT " INITIALIZING ";
13780 FOR I = 0 TO 27
13790 PRINT ". ";
13800 FOR J = 0 TO 4
13810 FOR K = 0 TO 11: STAT(I,J,K) = 0: NEXT K
13820 NEXT J
13830 NEXT I
13840 FOR J = 0 TO 13
13850 PRINT ". ";

```

```

13860         FOR I = 0 TO 44
13870             FOR K = 0 TO 5: ASSIGN(I,J,K) = 0: NEXT K
13880         NEXT I
13890     NEXT J
13900     RETURN
13910 REM
13920 REM **
13930 REM *****
13940 REM **
13950 REM **     SUBROUTINE TO INITIALIZE RANDOM NUMBER GENERATOR
13960 REM **
13970 REM
13980     XT = TIME: XT = XT - (INT(XT/100)*100)
13990     FOR I = 1 TO XT: X = RND: NEXT I
14000     X = X * 10000: RANDOMIZE X
14010     STDNT = STAT(0,0,0)
14020     RETURN
14030 REM
14040 REM **
14050 REM *****
14060 REM **
14070 REM **     SUBROUTINE TO RANDOMLY ALLOCATE CHARACTERISTICS TO SEMINARS
14080 REM **
14090 REM
14100     PRINT: PRINT "   ALLOCATING SLOTS TO SEMINARS   ";
14110     FOR J = 1 TO 4
14120         GOSUB 14870: KO = 1
14130         FOR I = 3 TO 27
14140             IF I/4 > KO THEN PRINT ". "; KO = KO + 1
14150             IF MIX = 1 THEN STAT(I,J,0) = ALOT(I,J,0)
14160             ALOT(I,0,0) = STAT(I,0,0)
14170             IF STAT(I,J,0) = 0 THEN GOTO 14390
14180             TA = STAT(I,J,0) / NS           :REM ** TEMPORARY ALLOCATION **
14190             ALOT(I,J,0) = STAT(I,J,0)
14200             IF TA < 1 THEN TA = 0: GOTO 14270
14210             FOR K = 1 TO LS - FS + 1
14220                 IF K = MS(1) - FS + 1 THEN GOTO 14260
14230                 IF K = MS(2) - FS + 1 THEN GOTO 14260
14240                 IF K = MS(3) - FS + 1 THEN GOTO 14260
14250                 ALOT(I,J,K) = INT(TA)       :REM ** GIVE SEMINAR SHARE **
14260             NEXT K
14270             IF STAT(I,J,0) - (NS * INT(TA)) = 0 THEN GOTO 14390
14280             ADJ = 0
14290             FOR K = 1 TO STAT(I,J,0)-(NS * INT(TA)) :REM ** GIVE REST **
14300                 ADJ = ADJ + 1
14310                 IF ADJ > 11 THEN ADJ = 1
14320                 IF ADJ = MS(1) - FS + 1 THEN GOTO 14300
14330                 IF ADJ = MS(2) - FS + 1 THEN GOTO 14300
14340                 IF ADJ = MS(3) - FS + 1 THEN GOTO 14300

```

```

14350             ALOT(1,J,ADJ) = ALOT(1,J,ADJ) + 1
14360             ALOT(0,J,ADJ) = ALOT(0,J,ADJ) + 1
14370             NEXT K
14380             IF MIX = 1 THEN STAT(1,J,0) = 0
14390             NEXT I
14400             ALOT(0,J,0) = STAT(0,J,0)
14410             TA = 25: ADJ = 0
14420             FOR K = 1 TO NS
14430                 ADJ = ADJ + 1
14440                 IF ADJ = MS(1) - FS + 1 THEN GOTO 14430
14450                 IF ADJ = MS(2) - FS + 1 THEN GOTO 14430
14460                 IF ADJ = MS(3) - FS + 1 THEN GOTO 14430
14470                 ALOT(0,J,ADJ) = INT(ALOT(0,J,0)/NS)
14480                 IF TA = 25 THEN TA = ALOT(0,J,0) - (ALOT(0,J,ADJ) * NS)
14490                 IF TA = 0 THEN GOTO 14520
14500                 ALOT(0,J,ADJ) = ALOT(0,J,ADJ) + 1
14510                 TA = TA - 1
14520             NEXT K
14530             STAT(0,J,0) = 0
14540             NEXT J
14550             PRINT
14560             RETURN
14570 REM
14580 REM **
14590 REM *****
14600 REM **
14610 REM ** SUBROUTINE TO RANDOMLY ASSIGN STUDENTS TO TEMPORARY SEMINARS **
14620 REM **
14630 REM
14640             RS = RND
14650             FOR LL = 1 TO LS - FS + 1
14660                 IF RS > LL/(LS - FS + 1) GOTO 14750
14670                 TSEM = FS + LL - 1: PS = LL
14680                 FOR R = 1 TO 3
14690                     IF TSEM = MS(R) THEN GOTO 14720
14700                 NEXT R
14710                 GOTO 14760
14720                 LL = LL + 1
14730                 IF LL > 11 THEN LL = 1
14740                 GOTO 14670
14750             NEXT LL
14760             IF MIX = 1 THEN KK = 0 ELSE KK = PS
14770             IF MIX = 1 THEN JJ = PS ELSE JJ = J
14780             IF TRONS > 1 THEN PRINT " * DEBUG: J=";J;" JJ=";JJ;" KK=";KK;" PS=";PS
14790             RETURN
14800 REM
14810 REM **
14820 REM *****
14830 REM **

```

```

14840 REM **      SUBROUTINE TO GET SEMINAR DATA FOR SQUADRON      **
14850 REM **                                                                 **
14860 REM
14870      FS = VAL(SQ$(J,2))                :REM ** FIRST SEMINAR IN SQ  **
14880      LS = VAL(SQ$(J,3))                :REM ** LAST SEMINAR IN SQ   **
14890      MS(1) = VAL(SQ$(J,4))             :REM ** MISSING SEMINAR IN SQ **
14900      MS(2) = VAL(SQ$(J,5))             :REM ** MISSING SEMINAR IN SQ **
14910      MS(3) = VAL(SQ$(J,6))             :REM ** MISSING SEMINAR IN SQ **
14920      NS = VAL(SQ$(J,7))                :REM ** # SEMINARS IN SQ    **
14930      TS = STAT(0,J,0)                  :REM ** TOTAL STUDENTS IN SQ **
14940      RETURN
14950 REM
14960 REM **                                                                 **
14970 REM *****
14980 REM **                                                                 **
14990 REM **      SUBROUTINE TO DETERMINE CURRENT SQUADRON ASSIGNMENT      **
15000 REM **                                                                 **
15010 REM
15020      FOR J = 0 TO 4
15030          IF VAL(ASQ$) = VAL(SQ$(J,1)) THEN GOSUB 14870: GOTO 15060
15040      NEXT J
15050      J = 0
15060      JJ = J
15070      IF TRONS > 1 THEN PRINT "* DEBUG:  FINDING CORRECT SQUADRON - J="; J
15080      RETURN
15090 REM
15100 REM **                                                                 **
15110 REM *****
15120 REM **                                                                 **
15130 REM **      SUBROUTINE TO FIND OFFSET FOR PRE-ASSIGNED STUDENTS      **
15140 REM **                                                                 **
15150 REM
15160      TSEM = 0
15170      IF MIX = 1 THEN TSEM = VAL(MIX1$)
15180      IF MIX = 2 THEN TSEM = VAL(MIX2$)
15190      IF MIX = 3 THEN TSEM = VAL(MIX3$)
15200      GOSUB 15290
15210      RETURN
15220 REM
15230 REM **                                                                 **
15240 REM *****
15250 REM **                                                                 **
15260 REM **      SUBROUTINE TO CHECK AGAINST MISSING SEMINARS              **
15270 REM **                                                                 **
15280 REM
15290      IF TSEM = 0 THEN GOTO 15370
15300      KK = TSEM - FS + 1
15310      IF KK >= 0 THEN GOTO 15340
15320      PRINT "*** ERROR *** SQUADRON ";ASQ$;" AND SEMINAR ";TSEM;" MISMATCH"

```

```

15330 PRINT " WILL REASSIGN TO NEW SEMINAR ": GOSUB 14560
15340 IF TSEM = MS(1) AND TSEM <> 0 THEN GOSUB 14640: GOTO 15370
15350 IF TSEM = MS(2) AND TSEM <> 0 THEN GOSUB 14640: GOTO 15370
15360 IF TSEM = MS(3) AND TSEM <> 0 THEN GOSUB 14640
15370 IF TRONS > 1 THEN PRINT "* DEBUG: TSEM=";TSEM;" FINDING CURRENT SMNR"
15380 RETURN
15390 REM
15400 REM **
15410 REM *****
15420 REM **
15430 REM ** SUBROUTINE TO CHECK ON PREVIOUSLY ASSIGNED STUDENTS **
15440 REM **
15450 REM
15460 IF SWAPFLG$ = "Y" GOTO 15480
15470 IF SEMI(KK) = 100 GOTO 15680
15480 MX1FLG = 0: MX2FLG = 0: MX3FLG$ = "N": SASN$ = "Y"
15490 CN = 0: MIX1CNT = 0: MIX2CNT = 0
15500 IF SWITCH = 0 THEN SEMI(KK) = 100
15510 IF TSEM = VAL(MIX1$) AND MIX <> 1 THEN GOTO 15680
15520 IF TSEM = VAL(MIX2$) AND MIX = 3 THEN GOTO 15680
15530 FOR AB = 1 TO ASSIGN(TSEM,0,0)
15540 IF MIX <> 1 AND ASSIGN(TSEM,AB,1) = VAL(MIX1$) THEN MX1FLG = 1
15550 IF MIX = 3 AND ASSIGN(TSEM,AB,2) = VAL(MIX2$) THEN MX2FLG = 1
15560 IF MX1FLG = 0 AND MX2FLG = 0 THEN GOTO 15620
15570 IF MX1FLG = 1 AND MX2FLG = 1 THEN MX3FLG$ = "Y": GOTO 15680
15580 IF SLFLAG$ = "Y" AND AB = 1 THEN GOTO 15680
15590 IF MX1FLG = 1 THEN MIX1CNT = MIX1CNT + 1
15600 IF MX2FLG = 1 THEN MIX2CNT = MIX2CNT + 1
15610 CN = CN + 1: MX1FLG = 0: MX2FLG = 0
15620 NEXT AB
15630 IF SLFLAG$ = "Y" GOTO 15690
15640 IF CN >= MAXSTDNTS THEN GOTO 15670
15650 IF SWITCH = 0 THEN SEMI(KK) = 0: SEMI(12) = 0
15660 GOTO 15690
15670 IF SWITCH = 0 THEN SEMI(KK) = CN + 300
15680 SASN$ = "N"
15690 IF TRONS > 1 THEN PRINT "* DEBUG: PREV=";SASN$;SLFLAG$;MX3FLG$;CN
15700 RETURN
15710 REM
15720 REM **
15730 REM *****
15740 REM **
15750 REM ** SUBROUTINE TO CHECK ALLOCATION RESTRICTIONS **
15760 REM **
15770 REM
15780 IF SWAPFLG$ = "Y" GOTO 15810
15790 IF SEMI(KK) > 300 THEN SASN$ = "N": GOTO 16000
15800 IF SEMI(KK) = 100 THEN SASN$ = "N": GOTO 16000
15810 SASN$ = "Y"

```

```

15820 GOSUB 13440
15830 FOR II = 0 TO 27
15840 IF (II = 1 OR II = 2) THEN GOTO 15990
15850 IF II = 0 GOTO 15870
15860 IF CHAR(II) = 0 THEN GOTO 15990
15870 IF STAT(II,JJ,KK) < ALOT(II,JJ,KK) THEN GOTO 15990
15880 FOR LL = 1 TO 11
15890 IF ALOT(II,JJ,LL) <= ALOT(II,JJ,KK) THEN GOTO 15950
15900 IF STAT(II,JJ,LL) >= ALOT(II,JJ,LL) THEN GOTO 15950
15910 TEMP = ALOT(II,JJ,KK)
15920 ALOT(II,JJ,KK) = ALOT(II,JJ,LL)
15930 ALOT(II,JJ,LL) = TEMP
15940 GOTO 15990
15950 NEXT LL
15960 IF SWITCH = 0 THEN SEMI(KK) = II + 200
15970 IF SWITCH = 0 AND II = 0 THEN SEMI(KK) = 100
15980 SASN$ = "N": GOTO 16000
15990 NEXT II
16000 RETURN
16010 REM
16020 REM **
16030 REM *****
16040 REM **
16050 REM ** SUBROUTINE TO ASSIGN TO NEW MIX
16060 REM **
16070 REM
16080 IF TSEM < 10 THEN Z = 1 ELSE Z = 2
16090 TSEM$ = MID$(STR$(TSEM),2,Z)
16100 IF TSEM < 10 THEN TSEM$ = "0"+MID$(TSEM$,1,1)
16110 RSET MIXX$ = TSEM$
16120 RSET PRIOR$ = STR$(PRIOR)
16130 IF MIX = 1 THEN LSET ASQ$ = SQ$(PS,1)
16140 SASN$ = "Y"
16150 RETURN
16160 REM
16170 REM **
16180 REM *****
16190 REM **
16200 REM ** SUBROUTINE TO ASSIGN SL & ASL TO EACH SEMINAR
16210 REM **
16220 REM
16230 PRINT: PRINT " ASSIGNING SL & ASL ( PRIORITY = 9 )": PRINT
16240 SLFLAG$ = "Y": ARFLAG$ = "N"
16250 SLASLF = 0: MAXSTDNTS = 1
16260 FOR J = 1 TO 4
16270 GOSUB 14870: SL(J) = NS: ASL(J) = NS
16280 NEXT J
16290 FOR J = 1 TO 44
16300 ASSIGN(J,0,0) = 2

```

```

16310     NEXT J
16320     FOR SSI = 1 TO STDNT
16330         SI = SSI
16340         GET #2,SI: GET #3,SI
16350         IF TRONS > 1 THEN PRINT "* DEBUG: RCRD="; SI; SNAME$
16360         IF MIX = 1 GOTO 16410
16370         IF VAL(RULE$(22)) = 9 AND (IO$ = "Y" OR IO$ = "y") GOTO 16400
16380         IF VAL(RULE$(19)) = 9 AND (SOS$ = "Y" OR SOS$ = "y") GOTO 16400
16390         GOTO 16410
16400         LSET MIX3$ = MIX1$
16410         GOSUB 15020                                :REM ** FIND CORRECT SQUADRON **
16420         GOSUB 14870                                :REM ** FIND SEMINAR DATA **
16430         GOSUB 15160                                :REM ** FIND OFFSET **
16440         LIS = 0
16450         IF TSEM > 0 THEN LIS = 1: KK = TSEM - FS + 1 :REM ** LOCK-IN **
16460         IOH$ = "
16470         IF IO$ = "Y" THEN IOH$ = " INTERNATIONAL OFFICER "
16480         IF IO$ = "y" THEN IOH$ = " INTERNATIONAL OFFICER "
16490         IF SOS$ = "Y" THEN IOH$ = " SOS INSTRUCTOR "
16500         IF SOS$ = "y" THEN IOH$ = " SOS INSTRUCTOR "
16510         IF SLASLF = 1 GOTO 17080                      :REM ** DONE **
16520         IF IO$ = "Y" OR IO$ = "y" THEN GOTO 17080
16530         IF SL$ = "Y" OR SL$ = "y" THEN GOTO 17080
16540         IF CC$ = "Y" OR CC$ = "y" THEN GOTO 17080
16550         IF SRO$ = "Y" OR SRO$ = "y" THEN GOTO 17080
16560         IF MIX < 3 AND (SOS$ = "Y" OR SOS$ = "y") THEN GOTO 17080
16570         IF MIX = 1 AND (USAF$ <> "Y" AND USAF$ <> "y") THEN GOTO 16800
16580         IF MIX = 1 AND (NOLINE$ = "Y" OR NOLINE$ = "y") THEN GOTO 16800
16590         IF TRONS > 1 THEN PRINT "* DEBUG: SL/ASL - LIS="; LIS; SNAME$
16600         SSL = 0
16610         FOR I = 0 TO 12: SEMI(I) = 299: NEXT I
16620         IF MS(1) <> 0 THEN SEMI(MS(1) - FS + 1) = 100
16630         IF MS(2) <> 0 THEN SEMI(MS(2) - FS + 1) = 100
16640         IF MS(3) <> 0 THEN SEMI(MS(3) - FS + 1) = 100
16650         IF LIS = 0 THEN GOSUB 14640                  :REM ** RANDOMLY ASSIGN **
16660         IF TRONS > 1 THEN PRINT "* DEBUG: TSEM="; TSEM;JJ;KK;SSL;FS;LIS
16670         IF SEMI(KK) = 100 AND LIS = 0 THEN GOTO 16650
16680         IF SSL = 1 GOTO 16810
16690         IF SL(JJ) = 0 GOTO 16800
16700         IF ASSIGN(TSEM,1,0) > 0 GOTO 16750
16710         SWITCH = 2: GOSUB 15460                      :REM ** CHECK CLASSMATES **
16720         IF SASN$ = "N" GOTO 16750
16730         SWITCH = 2: GOSUB 15780                      :REM ** CHECK CONSTRAINTS **
16740         IF SASN$ = "Y" GOTO 16960                    :REM ** ASSIGN AS THE SL **
16750         IF LIS = 1 THEN GOTO 16800                    :REM ** TRY TO USE AS ASL **
16760         SEMI(KK) = 100
16770         FOR I = 1 TO 11
16780             IF SEMI(I) = 299 GOTO 16650              :REM ** TRY ANOTHER SEMINAR **
16790         NEXT I

```

```

16800      SSL = 1: GOTO 16610
16810      IF ASL$ = "Y" OR ASL$ = "y" THEN GOTO 16830
16820      IF ASL(JJ) <> 0 GOTO 16840          :REM ** DONE WITH ASL'S? **
16830      IF LIS = 1 GOTO 17080 ELSE GOTO 17120
16840      IF ASSIGN(TSEM,2,0) > 0 GOTO 16900
16850      IF ASSIGN(TSEM,1,0) = 0 GOTO 16900
16860      SWITCH = 2: GOSUB 15460          :REM ** CHECK CLASSMATES **
16870      IF SASN$ = "N" GOTO 16900
16880      SWITCH = 2: GOSUB 15780          :REM ** CHECK CONSTRAINTS **
16890      IF SASN$ = "Y" GOTO 16960      :REM ** ASSIGN AS THE ASL **
16900      IF LIS = 1 GOTO 17080
16910      SEMI(KK) = 100
16920      FOR I = 1 TO 11
16930          IF SEMI(I) = 299 GOTO 16650      :REM ** TRY ANOTHER SEMINAR **
16940      NEXT I
16950      GOTO 17120
16960      IF SSL = 0 THEN ASG$ = "SL" ELSE ASG$ = "ASL"
16970      IF TRONS = 0 THEN GOTO 17000
16980      PRINT " ASSIGNING ";SNAME$;" # =";SI;" AS ";
16990      PRINT ASG$;" FOR SEMINAR ";TSEM;" "
17000      GOSUB 16080: GOSUB 17220          :REM ** UPDATE RECORDS **
17010      IF SSL = 0 THEN SL(JJ) = SL(JJ) - 1 ELSE ASL(JJ) = ASL(JJ) - 1
17020      FOR I = 1 TO 4
17030          IF SL(I) <> 0 GOTO 17070
17040          IF ASL(I) <> 0 GOTO 17070
17050      NEXT I
17060      SLASLF = 1
17070      GOTO 17120          :REM ** GET THE NEXT STUDENT **
17080      IF TSEM = 0 GOTO 17120          :REM ** CHECK FOR LOCK-INS **
17090      IF TRONS > 0 THEN PRINT " *LOCK IN* "; SNAME$; " TO "; TSEM; IOH$
17100      PRINT #1, " *LOCK IN* "; SNAME$; " TO "; TSEM; IOH$
17110      GOSUB 15020: GOSUB 14870: GOSUB 16080: GOSUB 17390
17120      NEXT SSI
17130      SLFLAG$ = "N": ARFLAF$ = "N"      :REM ** ALL SL'S & ASL'S ASSIGNED **
17140      RETURN
17150      REM
17160      REM **
17170      REM *****
17180      REM **
17190      REM ** SUBROUTINE TO UPDATE STUDENT RECORDS
17200      REM **
17210      REM
17220      RETEMP = ASSIGN(VAL(MIXX$),0,0)
17230      IF SSL = 1 GOTO 17270
17240      LSET SLX$ = "Y"          :REM ** UPDATE SL **
17250      ASSIGN(VAL(MIXX$),0,0) = 0
17260      GOTO 17290
17270      LSET ASLX$ = "Y"          :REM ** UPDATE ASL **
17280      ASSIGN(VAL(MIXX$),0,0) = 1

```

```

17290      GOSUB 17390
17300      ASSIGN(VAL(MIXX$),0,0) = RETEMP
17310      RETURN
17320  REM
17330  REM **
17340  REM *****
17350  REM **
17360  REM **   SUBROUTINE TO WRITE OUT STUDENT RECORDS
17370  REM **
17380  REM
17390      RSET PRIOR$ = STR$(PRIOR)
17400      ASSIGN(VAL(MIXX$),0,0) = ASSIGN(VAL(MIXX$),0,0) + 1
17410      ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),0) = SI
17420      IF MIX=2 THEN ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),1)=VAL(MIX1$)
17430      IF MIX=3 THEN ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),1)=VAL(MIX1$)
17440      IF MIX=3 THEN ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),2)=VAL(MIX2$)
17450      IF MIX=1 THEN ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),1)=VAL(MIXX$)
17460      IF MIX=2 THEN ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),2)=VAL(MIXX$)
17470      IF MIX=3 THEN ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),3)=VAL(MIXX$)
17480      ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),4) = PRIOR
17490      ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),5) = IR
17500      RECNUM(SI) = PRIOR + 50
17510      IF PRVFLG$ = "Y" THEN PREV$ = STR$(MAXSTDNTS): PRVFLG$ = "N"
17520      FSFLG$ = "N"
17530      PUT #2,SI: PUT #3,SI
17540      DELTA = + 1: GOSUB 13270
17550      IF TRONS < 2 GOTO 17620
17560      PRINT  "* DEBUG: ";STR$(SI);", ";SNAME$;MIXX$;". PRIOR=";PRIOR$;
17570      PRINT  ", ";SLX$;ASLX$;". ";STR$(ASSIGN(VAL(MIXX$),0,0));". ";
17580      PRINT  ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),1);
17590      PRINT  ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),2);
17600      PRINT  ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),3);
17610      PRINT  ASSIGN(VAL(MIXX$),ASSIGN(VAL(MIXX$),0,0),4)
17620      RETURN
17630  REM
17640  REM **
17650  REM *****
17660  REM **
17670  REM **   SUBROUTINE TO LOCATE ASSIGNED SKILLS TO SWITCH
17680  REM **
17690  REM
17700      SWTCH = 0
17710      FOR SLT = 1 TO 11
17720          IF TRONS > 1 THEN PRINT  "* DEBUG: SLT="; SLT; SWTCH
17730          IF TRONS > 1 THEN LPRINT  "* DEBUG: SLT="; SLT; SWTCH
17740          IF STAT(IR,JJ,SLT) > ALOT(IR,JJ,SLT) THEN GOTO 18580
17750          IF SEMI(SLT) < 200 OR SEMI(SLT) > 300 THEN GOTO 18590
17760          TEMP1 = SEMI(SLT) - 200
17770          TSEM = SLT + FS - 1

```

```

17780      SWITCH = 2: GOSUB 15460: SWITCH = 0
17790      IF MX3FLG$ = "Y" GOTO 18580
17800      FOR PTR = ASSIGN(TSEM,0,0) TO 3 STEP -1
17810          IF TRONS > 1 THEN PRINT "* DEBUG: PTR="; PTR; SEMI(SLT)
17820          IF TRONS > 1 THEN LPRINT "* DEBUG: PTR="; PTR; SEMI(SLT)
17830          IF ASSIGN(TSEM,PTR,4) > PRIOR + OMEGA THEN GOTO 18580
17840          SI = ASSIGN(TSEM,PTR,0)
17850          GET #2,SI: GET #3,SI
17860          GOSUB 13440
17870          IF OMEGA <> 0 GOTO 17900
17880          IF FSFLG$ = "Y" AND IR <> 0 AND CHAR(IR) <> 1 GOTO 18570
17890          IF FSFLG$ = "N" AND CHAR(TEMP1) <> 1 GOTO 18570
17900          DELTA = -1: KK = SLT: GOSUB 13270           :REM ** REMOVE **
17910          GET #2,SK: GET #3,SK
17920          GOSUB 13440
17930          IF SWCH = 0 GOTO 17970
17940          FOR YY = 3 TO 27
17950              IF VAL(RULE$(YY)) = 0 THEN CHAR(YY) = 0
17960          NEXT YY
17970          SWITCH = 2: KK = SLT
17980          GOSUB 15780: SWITCH = 0: TEMP$ = SNAME$: TMP$ = STN$
17990          GET #2,SI: GET #3,SI
18000          GOSUB 13440
18010          DELTA = +1: KK = SLT: GOSUB 13270
18020          IF FSFLG$ = "Y" GOTO 18040
18030          IF SASN$ = "N" GOTO 18570
18040          IF SWCH = 0 GOTO 18080
18050          FOR YY = 3 TO 27
18060              IF VAL(RULE$(YY)) = 0 THEN CHAR(YY) = 0
18070          NEXT YY
18080          SWAPFLG$ = "Y"
18090          FOR SMNR = FS TO LS
18100              IF TRONS > 1 THEN PRINT "* DEBUG: SMNR="; SMNR
18110              IF TRONS > 1 THEN LPRINT "* DEBUG: SMNR=";SMNR;FSFLG$;FS
18120              IF SMNR = TSEM GOTO 18550
18130              IF SMNR = MS(1) GOTO 18550
18140              IF SMNR = MS(2) GOTO 18550
18150              IF SMNR = MS(3) GOTO 18550
18160              IF (MIX = 2 OR MIX = 3) AND SMNR = VAL(MIX1$) GOTO 18550
18170              IF MIX = 3 AND SMNR = VAL(MIX2$) THEN GOTO 18550
18180              XX = SMNR - FS + 1
18190              SWITCH = 2: KK = XX
18200              GOSUB 15780: SWITCH = 0
18210              IF TRONS > 1 THEN LPRINT "* DEBUG: ALOC=";SASN$;KK;SNAME$
18220              IF FSFLG$ = "Y" GOTO 18250
18230              IF SASN$ = "N" GOTO 18550
18240              TSEM = SMNR
18250              SWITCH = 2: GOSUB 15460: SWITCH = 0: TSEM = SLT + FS - 1
18260              IF TRONS > 1 THEN LPRINT "* DEBUG: PRV=";SASN$;MX3FLG$;CN

```

```

18270      IF MX3FLG$ = "Y" GOTO 18550
18280      IF SASN$ = "N" THEN MXFLG$ = "Y": GOTO 18550
18290      IF TRONS = 0 GOTO 18320
18300      PRINT " *** EXCHANGING *** ----- "; TEMP$
18310      PRINT "                WITH ----- "; SNAME$
18320      IF LOGFLG$ <> "Y" THEN GOTO 18350
18330      PRINT #1,"*** EXCHANGING *** (STU # ";TMP$;") "; TEMP$
18340      PRINT #1,"..... WITH (STU # ";STN$;") "; SNAME$
18350      FOR YY = PTR TO 13
18360          TEMPA = ASSIGN(TSEM,PTR,5)
18370          FOR R = 0 TO 5
18380              IF YY = 13 THEN ASSIGN(TSEM,13,R) = 0: GOTO 18400
18390              ASSIGN(TSEM,YY,R) = ASSIGN(TSEM,YY+1,R)
18400              IF R=5 AND ASSIGN(TSEM,YY,0) = 0 THEN GOTO 18430
18410          NEXT R
18420      NEXT YY
18430      ASSIGN(TSEM,0,0) = YY - 1
18440      TSEM = SMNR: IR = TEMPA
18450      IF LOGFLG$ <> "Y" THEN GOTO 18470
18460      PRINT #1,"ASSIGNING ";SNAME$;"# =";SI;" TO SEMINAR ";TSEM
18470      IF TRONS = 0 THEN GOTO 18490
18480      PRINT " ASSIGNING ";SNAME$;" # =";SI;" TO SEMINAR ";TSEM
18490      GOSUB 16080: GOSUB 17390
18500      IR = INDEX                      :REM ** REMOVE **
18510      DELTA = -1: KK = SLT: GOSUB 13270
18520      TSEM = SLT + FS - 1
18530      SWAPFLG$ = "N"
18540      GOTO 18610
18550      NEXT SMNR
18560      SWAPFLG$ = "N"
18570      NEXT PTR
18580      SEMI(SLT) = 100
18590      NEXT SLT
18600      SEMI(12) = 2
18610      SWCH = 0
18620      RETURN
18630 REM
18640 REM **
18650 REM *****
18660 REM **
18670 REM ** SUBROUTINE TO PROCESS REMAINING STUDENTS
18680 REM **
18690 REM
18700      NP = STAT(IR,0,0)
18710      PRINT
18720      PRINT " ASSIGNING REMAINING "; RL$;" STUDENTS ";
18730      PRINT " ( PRIORITY = "; PRIOR;" )"
18740      PRINT
18750      NP = STAT(IR,0,0)

```

```

18760 IF NP = 0 THEN GOTO 19820
18770 ARFLAG$ = "N":SLFLAG$ = "N":MXFLG$ = "N":PRVFLG$ = "N":FSFLG$ = "N"
18780 FOR SSI = 1 TO STDNT
18790 SI = SSI
18800 IF RECNUM(SI) <> 0 THEN GOTO 19810
18810 GET #2,SI: GET #3,SI
18820 LOGFLG$ = "N": OMEGA = 0
18830 IF IR = 0 THEN GOTO 18860
18840 GOSUB 13440
18850 IF CHAR(IR) = 0 THEN GOTO 19810
18860 GOSUB 15020 :REM ** FIND SQUADRON DATA **
18870 GOSUB 14870 :REM ** FIND SEMINAR DATA **
18880 GOSUB 15160 :REM ** FIND OFFSET **
18890 FOR I = 0 TO 12: SEMI(I) = 0: NEXT I
18900 IF MS(1) <> 0 THEN SEMI(MS(1) - FS + 1) = 100
18910 IF MS(2) <> 0 THEN SEMI(MS(2) - FS + 1) = 100
18920 IF MS(3) <> 0 THEN SEMI(MS(3) - FS + 1) = 100
18930 GOSUB 14640 :REM ** RANDOMLY ASSIGN **
18940 IF TRONS > 1 THEN PRINT "* DEBUG: TSEM="; TSEM; JJ; KK
18950 SWITCH = 0: GOSUB 15460 :REM ** CHECK PREV CLASSMATES **
18960 IF SASN$ = "N" THEN GOTO 18990
18970 SWITCH = 0: GOSUB 15780 :REM ** CHECK ON CONSTRAINTS **
18980 IF SASN$ = "Y" THEN GOTO 19740
18990 SEMI(0) = 0: SEMI(12) = 0
19000 FOR I = 1 TO 11
19010 IF SEMI(I) = 0 THEN GOTO 18930 :REM ** TRY AGAIN **
19020 IF SEMI(I) > 300 THEN SEMI(0) = 1: GOTO 19040
19030 IF SEMI(I) > 200 THEN SEMI(12) = 1
19040 NEXT I
19050 IF TRONS < 2 GOTO 19090
19060 PRINT "* DEBUG: SEMI=";
19070 FOR I = 0 TO 12: PRINT SEMI(I);: NEXT I
19080 PRINT SI; SK: PRINT
19090 IF SEMI(12) = 1 THEN GOTO 19680
19100 IF SEMI(0) = 1 GOTO 19140
19110 IF PRVFLG$ = "N" GOTO 19130
19120 PREV$ = STR$(MAXSTDNTS): PRVFLG$ = "N": GOTO 19340
19130 IF MXFLG$ = "Y" THEN GOTO 19200 ELSE GOTO 19330
19140 SEMI(0) = 0
19150 FOR I = 1 TO 11
19160 IF SEMI(I) = 100 GOTO 19190
19170 IF MAXSTDNTS <> SEMI(I) - 300 GOTO 19190
19180 SEMI(I) = 0
19190 NEXT I
19200 MAXSTDNTS = MAXSTDNTS + 1
19210 IF MXFLG$ = "Y" THEN MXFLG$ = "N": GOTO 18890
19220 IF MAXSTDNTS <= VAL(PREV$) GOTO 18890
19230 IF MAXSTDNTS > 9 THEN MAXSTDNTS = VAL(PREV$): GOTO 18990
19240 PRVFLG$ = "Y": LOGFLG$ = "Y"

```

```

19250 PRINT #1, "*** WARNING *** UNABLE TO ASSIGN ( #"; SI;" ) "; SNAME$
19260 PRINT #1, " TO SEMINAR WITH LESS THAN";
19270 PRINT #1, MAXSTDNTS - 1;"PREVIOUS CLASSMATES"
19280 IF TRONS = 0 THEN GOTO 18890
19290 PRINT " *** WARNING *** UNABLE TO ASSIGN ( #"; SI;" ) "; SNAME$
19300 PRINT " TO SEMINAR WITH LESS THAN";
19310 PRINT MAXSTDNTS - 1;"PREVIOUS CLASSMATES"
19320 GOTO 18890
19330 IF LOGFLG$ = "Y" THEN GOTO 19390
19340 IF TRONS = 0 THEN GOTO 19360
19350 PRINT " *** FORCING ASSIGNMENT FOR "; SNAME$;"(STU # "; STN$;" )"
19360 PRINT #1, "*** FORCING ASSIGNMENT FOR "; SNAME$
19370 PRINT #1, ". (STU # "; STN$;" ) - RULE FOR "; STAT$(IR)
19380 LOGFLG$ = "Y"
19390 SWCH = 0: KK = 0
19400 TEMP = 15: LL = 0
19410 FOR PTR = 1 TO 11
19420 IF SEMI(PTR) = SWCH GOTO 19470
19430 IF MIX <> 1 AND VAL(MIX1$) = PTR + FS - 1 THEN GOTO 19470
19440 IF MIX = 3 AND VAL(MIX2$) = PTR + FS - 1 THEN GOTO 19470
19450 IF STAT(0,JJ,PTR) >= ALOT(0,JJ,PTR) GOTO 19470
19460 IF TEMP > STAT(0,JJ,PTR) THEN TEMP = STAT(0,JJ,PTR): LL = PTR
19470 NEXT PTR
19480 KK = LL: TSEM = KK + FS - 1
19490 IF TRONS > 1 THEN PRINT"* DEBUG: TSEM";TSEM;KK;SWCH;FSFLG$;OMEGA
19500 IF FSFLG$ = "N" GOTO 19600
19510 IF KK <> 0 THEN GOTO 19670
19520 IF OMEGA + PRIOR < 5 THEN OMEGA = OMEGA + 1: GOTO 18890
19530 PRINT #1, "-----"
19540 PRINT #1, "*** CAN'T ASSIGN *** "; SNAME$;" ( # "; STN$;" )"
19550 PRINT #1, "-----"
19560 PRINT " -----"
19570 PRINT " *** CAN'T ASSIGN *** "; SNAME$;" ( # "; STN$;" )"
19580 PRINT " -----"
19590 GOTO 19810
19600 IF SWCH > 1 THEN FSFLG$ = "Y": SWCH = 0: GOTO 18860
19610 IF TEMP = 15 THEN SWCH = SWCH + 1: GOTO 19410
19620 IF STAT(IR,JJ,KK) >= ALOT(IR,JJ,KK) THEN SEMI(KK)=SWCH: GOTO 19400
19630 SEMI(KK) = SWCH: GOSUB 15460
19640 IF MX3FLG$ = "Y" THEN SEMI(KK) = SWCH: GOTO 19400
19650 IF SWCH > 0 GOTO 19670
19660 IF SASN$ = "N" GOTO 19400
19670 SWCH = 0: GOTO 19740
19680 SK = SI: SWCH = 0
19690 GOSUB 17700 :REM ** TRY EXCHANGING **
19700 SI = SK
19710 GET #2,SI: GET #3,SI
19720 GOSUB 13440
19730 IF SEMI(12) = 2 THEN GOTO 18990

```

```

19740      IF LOGFLG$ <> "Y" THEN GOTO 19770
19750      PRINT #1, "ASSIGNING ";SNAME$;" # =";SI;" TO SEMINAR ";
19760      PRINT #1, TSEM; "      "
19770      IF TRONS = 0 THEN GOTO 19800
19780      PRINT "  ASSIGNING ";SNAME$;" # =";SI;" TO SEMINAR ";
19790      PRINT TSEM; "      "
19800      GOSUB 13440: GOSUB 16080: GOSUB 17390      :REM ** UPDATE RECORDS **
19810      NEXT SSI
19820      RETURN
19830 REM
19840 REM **
19850 REM *****
19860 REM **
19870 REM **  SUBROUTINE TO RANDOMLY ALLOCATE CHARACTERISTICS TO SQUADRONS **
19880 REM **
19890 REM
19900      PRINT "  ALLOCATING SLOTS TO SQUADRONS  ";
19910      GOSUB 14870
19920      ADJ = 0
19930      FOR I = 3 TO 27
19940          PRINT ". ";
19950          IF STAT(I,0,0) = 0 THEN GOTO 20070
19960          TA = STAT(I,0,0) / 4      :REM ** TEMPORARY ALLOCATION **
19970          IF TA < 1 THEN TA = 0: GOTO 20010
19980          FOR J = 1 TO 4
19990              ALOT(I,J,0) = INT(TA)      :REM ** GIVE SQUADRON FAIR SHARE **
20000          NEXT J
20010          IF STAT(I,0,0) - (4 * INT(TA)) = 0 THEN GOTO 20070
20020          FOR L = 1 TO STAT(I,0,0) - (4 * INT(TA))      :REM ** GIVE REST **
20030              ALOT(I,ADJ,0) = ALOT(I,ADJ,0) + 1
20040              ADJ = ADJ + 1
20050              IF ADJ > 4 THEN ADJ = 1
20060          NEXT L
20070      NEXT I
20080      PRINT
20090      RETURN
20100 REM
20110 REM **
20120 REM *****
20130 REM **
20140 REM **  SUBROUTINE TO OUTPUT RESULTS TO SMART IN ASCII **
20150 REM **
20160 REM
20170      CLS: LOCATE 2,25: PRINT "MIXER OUTPUTTING TO SMSS": PRINT: PRINT
20180      STDNT = STAT(0,0,0)
20190      OPEN "O", #1, "STUOUT.ASC"
20200      KO = 1: PRINT "  OUTPUTTING ";
20210      FOR SSI = 1 TO STDNT
20220          IF SSI/10 >= KO THEN PRINT ". ";: KO = KO + 1

```

```

20230      SI = SSI
20240      GET #2,SI: GET #3,SI
20250      OTREC$ = CHR$(34)+SNAME$+CHR$(34)+", "
20260      OTREC$ = OTREC$+CHR$(34)+STN$+CHR$(34)+", "
20270      OTREC$ = OTREC$+CHR$(34)+ASQ$+CHR$(34)+", "
20280      OTREC$ = OTREC$+CHR$(34)+MIXX$+CHR$(34)+", "
20290      OTREC$ = OTREC$+CHR$(34)+PRIOR$+CHR$(34)+", "
20300      OTREC$ = OTREC$+CHR$(34)+SLX$+CHR$(34)+", "
20310      OTREC$ = OTREC$+CHR$(34)+ASLX$+CHR$(34)+", "
20320      PRINT #1,OTREC$
20330      NEXT SSI
20340      PRINT #1,CHR$(26)
20350      CLOSE: RETURN
20360 REM
20370 REM **
20380 REM *****

```

Appendix Four

SMSS.BAT Program Listing

SMSS.BAT is an MS-DOS batch file. When the user enters "smss" to start the system, this file of MS-DOS commands is called. Its major function is to pass control between SMART and the mixer program.

SMSS.BAT

```
echo off
echo Calling SMART--Standby Please
rem SMSS.BAT--this is the DOS batch file used to call SMSS and control
rem entry and exit between SMART and the BASIC program, MIXER
rem
rem After operator enters "smss" from dos, entry is here
rem First call the SMART datamanager and project file smss
smart d -psmss
:smloop
rem Return from SMART either because the user hit Esc at the main menu
rem or because the user has chosen to call the mixer.
rem Existence of file env.asc will be used to test for which.
if not exist env.asc goto smexit
echo call mixer
mixer
rem do test for error exit
if exist error.sms goto smerr
echo Calling SMART--Standby
smart d -psmss
rem return could be from exit or mixer call. go back and check
goto smloop
:smerr
echo error exit from mixer
:smexit
echo exit from smss
```

END

DATE

FILMED

6-88

DTIC