NPS52-81-007

NAVAL POSTGRADUATE SCHOOL

Monterey, California

A PROTOTYPE PROGRAM FOR TARGET INFORMATION

Ronald J. Coulter

June 1981

Approved for public release; distribution unlimited.

The work reported herein was supported by the Microcomputer Laboratory, Department of Computer Science, Naval Postgraduate School, Monterey, California.

This report was prepared by:

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS52-81-007 / 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* A Prototype Program for Target Information	5. TYPE OF REPORT & PERIOD COVERED Technical Report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Ronald J. Coulter LTCOL USMC	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940	12. REPORT DATE June 1981
	13. NUMBER OF PAGES 128
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*	15. SECURITY CLASS. *(of this report)* UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Microcomputer, User interface, Target Information, Data Base, FSCC (Fire Support Coordination Center), Fire Support Coordination, Marine Corps, UCSD Pascal, Amphibious Operations

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This thesis presents the specification, design and implementation of a prototype microcomputer system for the target information section of the Marine Corps fire support coordination center. Currently, the target information section uses a series of index cards, handwritten lists, acetate covered battle maps and grease pencils to perform the target information functions.

The thesis examines and analyzes these functions in detail and proposes

20.

a solution in the form of a system, data base and interactive user design. The resultant Microcomputer System for Traget Information (MISTI) employs an ALTOS Z-80 microcomputer, the UCSD Pascal operating system, a user friendly interface and data base technology. It is proposed as an interim system until the Marine Integrated Fire and Air Support System (MIFASS) becomes operational.

TABLE OF CONTENTS

PREFACE

The enclosed technical report contains a [illegible] source code listing of a microcomputer program for the management of target information in the Marine Corps Fire Support coordination center. It is issued in conjunction with the masters thesis entitled <u>A Microcomputer System for Target Information in the Fire Support Coordination Center: A Data Base Approach</u> by the author. Details for system specification, design and implementation can be found in this thesis.

A PROTOTYPE PROGRAM FOR TARGET INFORMATION

Introduction

More and more of the applications of modern amphibious warfare have turned to computerized solutions, from real-time combat systems to the data bases that control the men, materiel and resources needed to wage war. The products of the technological explosion have enabled the Navy-Marine Corps amphibious team to do more, to do it faster and to do it with a degree of efficiency and accuracy previously unobtainable.

This evolution of modern technology has not yet reached the Marine Corps tactical command posts established on the beachhead. The target information section of the landing force fire support coordination center (FSCC) plays a signficant role in the conduct of effective coordination of tactical air, artillery and naval gunfire support on targets of high priority. Yet the target information officer and his staff accomplish their important task by the use of index card files, cross-reference files, hand written lists of targets and colored grease pencils on acetate-covered tactical maps. This method is time consuming, slow in response to inquires about target information, tedious and difficult to maintain in a current status and does not provide information in a sufficiently timely and accurate manner. It is 40 year old technology in the age of computers.

The requirement to automate many of the functions of the tactical command post has been identified and the command post of the future is being planned for and developed now. Until it arrives, there is a need to provide an interim capability to the landing force. An automated solution to the target information function will simplify the task of the target information section considerably, will provide rapid, accurate and timely target information to the members of the FSCC, and can be made operational now, five full years before the planned introduction of the computerized command post.

This report contains a prototype program for target information which will improve the operational capability of the landing force FSCC and show that the implementation of a suitable and effective target information system is possible. This implementation and design of a working prototype will increase operational effectiveness immediately as well as provide a testbed and learning model for the future automated command post. The prototype is designed to perform all the duties and functions of the target information section as currently stated in doctrinal publications. The interim system will hopefully contribute to the development of the future system and identify areas of concern and improvement before the future Marine Corps system becomes operational.

Background

An important aspect of amphibious fire support coordination (the planning and execution of tactical air, artillery and naval gunfire support so that targets are adequately covered by a suitable weapon or group of weapons) is the function of target information. One of the major duties of the fire support coordinator, that member of the landing force staff responsible for coordination of fire support, is to ensure that the fire support coordination center receives and disseminates available target information to all staff sections and commands requiring the information. He also must work closely with the target information officer and the commander and his staff in the selection of targets and assignment of classification and attack priorities.

Target information is the direct application of combat intelligence to fire support and is a key to the proper employment of supporting arms in conjunction with each of the plans of the amphibious operation. Effective fire support coordination and the planning of amphibious operations generate a continuing requirement for target acquisition, dissemination, evaluation and recommendation for attack.

To accomplish this important task, the commander of the amphibious task force assigns a target intelligence officer to the supporting arms coordination center (SACC). This officer operates the target information center (TIC) and works closely with the air intelligence officer, the landing force targeting representatives and the supporting arms coordinator. The commander of the landing force has a target information officer (TIO) who operates the target information section (TIS) as an integral part of the landing force fire support coordination center and a target intelligence officer who functions in the landing force intelligence center.

The Navy staff uses a computerized target information system which is part of the shipboard Amphibious Support Information System (ASIS) and maintains the list of targets as part of a data base. Target information operations in the SACC are thus computerized and, while the ASIS target system is not the most modern of data base systems, it is efficient, effective and fast. When the functional responsibility for maintaining targets is passed ashore to the landing force TIO, the computer system is replaced by an index card filing system, which , while effective, is neither fast nor efficient by comparison. Additionally, the index card system lends itself to inaccuracies and omissions in target data, particularly when the information must be maintained in a timely manner. The tactical requirement for accurate and timely target information is no less critical or important when the landing force is on the beach, yet the system to accomplish this task is antiquated and cumbersome.

The staff of the TIS manually transfers the target information data contained in the ASIS data base to 5 by 8 inch target cards. After duplicating the entire target file, the TIS must construct a cross reference file to list the target by grid location and a cross-index file to keep track of certain types of targets. In addition to the target cards, the TIS also makes up lists of particular categories of targets which may be of interest or value to members of the FSCC.

The TIS obtains intelligence information from landing force and supporting arms agencies, converts this to target information and enters the information into the target card files. The information is made available to the supporting arms representatives in the FSCC and, based on the TIO's recommendations, a decision is made when and how to attack a particular target. Results of attacks on targets, front line reports and intelligence information are used to refine the target list and delete or deprioritize those targets that present a diminished threat to the landing force.

Access to specific information from the target list (for example, more than one category of the cross-index files) requires physically searching through each list and constructing sub-lists to determine the appropriate information. The constant availability of timely and accurate target information is required for the effective employment of supporting arms and planning of fire support. The TIS plays a key role in providing this information and the constant process of adding to the target list, selecting targets for attack and deleting targets once neutralized is performed by the TIS staff using the target card file.

One of the most complex aspects of modern amphibious warfare is the control and coordination of supporting arms particularly in the transition of responsibility from the Navy in amphibious ships to the Marine Corps combat units ashore. The grease pencils, map boards and field radios that have served Marines so well since the days of Guadalcanal will, in the future, be eclipsed by the automated system called the Marine Integrated Fire and Air Support System (MIFASS).

MIFASS is part of the Marine Corps integrated command and control system called MTACCS (Marine Tactical Command and Control Systems), a collection of eight major systems which will give the Marines a capability of exercising real-time command and control of combat forces in the post-1980 time frame. MIFASS is designed to perform the functions of the fire support coordination center, (FSCC) the direct air support center (DASC) and, to a degree, the artillery fire direction center (FDC) at one central location called the Fire and Air Support Center (FASC).

It is a distributed processing system in which microcomputers control interactive display devices, manage data bases, perform computational tasks and drive printers to provide hard-copy records of messages and operator decisions. It is currently in full scale engineering development with an initial operational capability planned for the 1986-1987 time frame. MIFASS addresses the requirement for target information by providing the TIO with a digital display device which will have both a graphical representation of the target on a battle map and a video screen for alphanumeric display of target information.

Statement of the Problem

An automated solution to the target information function will not be realized until the introduction of the MIFASS computers into the Fleet Marine Forces. Until such time as the system is delivered, the target information function of the FSCC is tied to the current doctrine and the target card filing system.

In this report, an interim solution to the problem of automating the target information function of the FSCC is presented. It computerizes those basic functions of the TIS in a simple, inexpensive and effective manner. It simplifies the tasks of the TIS, provides a mechanism for rapid and accurate retrieval of target information and could improve the operational capability of the FSCC.

Nature of the Solution

The design task is broken down into three distinct parts, each of which is influenced by the dual constraints of a microcomputer environment and a friendly user interface. The design is specifically addressed in the thesis which is supported by this program listing report.

The design of the physical and logical data base is influenced by the desire to have a simple yet sufficiently informative data model, a rapid, real-time response and a restricted, single application system. The system design is influenced by the microcomputer environment which restricts the user both in main memory space and the speed of access to secondary storage and the requirement for an effective interactive system for a non-sophisticated user.

The design of the software to implement both the data base and the system is overwhelmingly influenced by the requirement that the system support real-time, interactive processing of a casual, non-programmer. Termed "Marine proof" in the vernacular, it requires a sophisticated interface employing user friendly dialogue techniques to ensure that the operation is simple and efficient. For this reason, and to facilitate system portability, a microcomputer compatable high level programming language (UCSD Pascal) is used.

The report is divided into two sections. The first is the source code listing, by module, of the Microcomputer System for Target Information (MISTI) program. The second is a listing of the text files used in the interactive user interface which complements much of the prototype program. The reader is referred to the Naval Postgraduate School masters thesis [1] for additional details on the specifications, design and implementation of the system. This report is issued in conjunction with and as a key element of the thesis.

----

1. Coulter, R.J., A Microcomputer System for Target Information in the Fire Support Coordination Center: A Data Base Approach, Masters Thesis, June 1981.

REFERENCES

Bowles, K. L., *Microcomputer Problem Solving Using Pascal*, Springer-Verlag, 1977.

Conrad, T., *Pascal--A High-level Language for Minis and Micros*, Datamation, July 1976.

Coulter, R. J., *A Microcomputer System for Target Information in the Fire Support Coordination Center: A Data Base Approach*, Masters Thesis, Naval Postgraduate School, June 1981.

Dahmke, M., *The Altos ACS 8000 Single Board Computer*, Byte Magazine, McGraw Hill, v. 5, no. 11, Nov 1980.

Engle, S. E. and Granda, R. E., *Guidelines for Man/Display Interface*, IBM Technical Report, 1975.

Fleet Marine Force Manual (FMFM) 7-1, United States Marine Corps, *Fire Support Coordination*, United States Government Printing Office, 1975.

Grogono, P., *Programming in Pascal*, Addison-Wesley, 1978.

Institute for Information Systems, University of California at San Diego, *UCSD (Mini-Micro Computer) Pascal*, Release Version 1.4, 1978.

Jensen, K. and Wirth, N., *Pascal: User Manual and Report*, 2nd ed., Springer-Verlag, 1974.

Lewis, T. G. and Smith, M. Z., *Applying Data Structures*, Houghton Mifflin Company, 1976.

Headquarters, United States Marine Corps, *Marine Tactical Command and Control Systems (MTACCS) Master Plan*, Washington, D. C., October 1979.

Marine Corps Tactical Systems Support Activity (MCTSSA), *MIFASS Specifications*, ELEX-[illegible]-280C, 1981.

Martin, J., *Design of Man-Computer Dialogue*, Prentice Hall, 1973.

Naval Warfare Publication (NWP 22-2), Department of the Navy, *Supporting Arms in Amphibious Operations*, Naval Warfare Publications, 1978.

Senan, A., and Sinombing, T. M., Database Management System for Microcomputers, Masters Thesis, Naval Postgraduate School, 1979.

Shneiderman, B., Software Psychology: Human Factors in Computer and Information Systems, Winthrop Publications Inc., 1980.

Smith, L. B., The Use of Interactive Graphics to Solve Numerical Problems, Communications of the ACM, 1970.

Snodgrass, R., A Sophisticated Microcomputer User Interface, Procedings of the Third Symposium on Small Systems, 1982.

APPENDIX--A

The source code listing for the MISTI system is organized by functional module. The list is not a compiled listing but is separate by function and by module. In an effort to decrease user confusion, the following outline shows the logical organization of the program. The segment procedures are marked by a *.

```
                          .....GLOBALS
                          :
                          :....QUERY*             ....BDA*
                          :                       :
                          :....TARGET*........:...TGTPROCS
                          :                       :
    INTERFACE.............:                       :
                          :                       :...ADDTARGET
                          :....INIT*              :
                          :                       :...CHANGE
                          :....INFORM*            :
                          :                       :...TARGET
                          :
                          :....UTILITY*
```

The Interface module contains include statements which instruct the compiler to compile the program in the proper logical order. The segment procedures are the first procedures to be compiled. This necessitates the declaring of many of the system routines in the beginning of the listing. The UCSD Pascal include statement allows the user to identify the volume name as well as the file name. "Store" is the name of the volume which contains the source code, thus, it appears in the include commands.

```
*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*)

*          MAIN PROGRAM.....INTERFACE          *)

rogram interface (input,output);


      (*$Istore:globals.text*)


rocedure clear; forward;         procedure delay; forward;
rocedure prompt; forward;        procedure spacebar; forward;
rocedure select; forward;        procedure returnbar; forward;
rocedure menuerror; forward;     procedure halt; forward;
rocedure error1; forward;        procedure error2; forward;
rocedure lines(y : integer); forward;
rocedure loadmsg; forward;
rocedure getfile( filename : string); forward;



      (*$Istore:query.text*)
      (*$Istore:bda.text*)
      (*$Istore:tgtprocs.text*)
      (*$Istore:addtarget.text*)
      (*$Istore:change.text*)
      (*$Istore:target.text*)
      (*$Istore:init.text*)
      (*$Istore:inform.text*)
      (*$Istore:utility.text*)


         procedure clear;

             begin
               write(chr( 31 ));
             end;


         procedure prompt;

              begin
                write(prompter);
              end;

         procedure spacebar;

              begin
                writeln(spacebr);
                prompt;
                repeat
                read(ch);
                until ch = ' ';
              end;
```

```
  procedure select;

       begin
         writeln(selectop);
         prompt;
       end;


  procedure lines;

       var   yy : integer;

       begin
         for yy := 1 to y do
         writeln;
       end;


procedure returnbar;

      begin
        writeln(returner);
        prompt;
        repeat
          read(ch);
        until eoln(input);
      end;


procedure loadmsg;

begin
  lines(3);
  write(chr(14));
  writeln('              PROGRAM BEING READ IN.....PLEASE WAIT')
  write(chr(24));
  writeln(chr(29));
end;


   procedure delay;

      var
        x, y, z : integer;

      begin
        z := 0;
        x := 0 ;
        repeat
          for y := 1 to 100 do
            z := z + 1 ;
          x := x + 1 ;
          until x = 10 ;
        end;


     procedure halt;

        var
           z : integer;

        begin
          lines(2);
          write('     System halting...');
          for z := 1 to 10 do
          begin
            delay;
            write(dot);
          end;
        end;


    procedure error1;

    begin
      lines(1);
      writeln('          The proper format is a number from the');
      writeln('          options listed above. Please press the');
      writeln('          RETURN key and reenter your choice.');
      lines(1);
      prompt;
    end;


     procedure menuerror;

     begin
       lines(1);
       if eoln (input) then
       begin
         error1;
         readln;
       end
       else error1;
       readln;
     end;


     procedure error2;


begin
  lines(3);
```

```
    writeln('        The target identifier does not currently');
    writeln('        exist in the target file. Please reenter');
    writeln('        the target identifier.');
    lines(1);
    numbout := numbout + 1;
    if numbout = 3 then
    begin
      lines(1);
      writeln('        To leave this procedure, type a 0');
      writeln('        followed by pressing the RETURN key.');
      lines(1);
      numbout := 0 ;
    end;
  end;



      procedure getfile;

      (*  filename : string  declared forward   *)

      var   buffer : string;
            usermessage : text;

      begin
        reset(usermessage,filename);
        repeat
          begin
            readln(usermessage,buffer);
            writeln(buffer);
          end;
        until eof(usermessage);
        close(usermessage,lock);
      end;


(*.....................................................*)


      procedure buildempty;

      var  j : integer;

   begin
     with emptyTrec do
       begin
         tnum := '000000';
         alt := '     ';
         loc := '00000000';
         acc := ' ';
         class := ' ';
         pri := ' ';
         ttype := ' ';
      sa := ' ';
      stat := ' ';
      desc := '                                                    ';
      rem := desc;
      maprefer := '                     ';
      sour := maprefer;
      photonum := '                ';
      DTGact := '          ';
      photocord := loc;
      for i := 1 to numbervar do
        flag[i] := off;
      for i := 1 to 3 do
      begin
        BDA[i].DTGsurv := DTGact;
        BDA[i].fireunit := '        ';
        BDA[i].ntrnds := '           ';
        BDA[i].damrep := ' ';
        BDA[i].damass := ' ';
        BDA[i].BDAtext := desc;
      end;
    end;
  with emptyGrec do
    begin
      tnum := '        ';
      alt := '      ';
      loc := '           ';
      class := ' ';
      pri := ' ';
      acc := ' ';
      ttype := ' ';
      sa := ' ';
      desc := '                       ';
    end;
  end;




           procedure buildmap;

var   j : integer;

begin
  i := 0;
  j := 0;
  while not eof(target) do
    begin
      seek(target,i);
      get(target);
      tgtmap[i] := target^.tgtrec.tnum;
      i := i + 1;
      j := j + 1;
      if j = 10 then
```

```
          begin
            write(dot);
            j := 0;
          end;
        end;
      end;



          procedure openfiles;

    var   io : integer;

    begin
      filecheck := false;
      clear;
      lines(6);
      loadmsg;
      lines(4);
      write('      Opening file...');
      write(dot);
      (*$I-*)
      reset (target,'#5:targetfile.data');
      (*$I+*)
      io := ioresult;
      if io in [4,5,9] then
      begin
        clear;
        writeln(chr(7));
        lines(8);
        writeln(chr(14));
        writeln('           ***   NO DISK IN DRIVE  B  ***');
        write(chr(24));
        writeln(chr(29));
        lines(5);
        writeln('  Insert TARGET diskette in drive B and restart system
        halt;
        getout := true;
        exit(openfiles);
      end;
      if io <> 0 then
      begin
        filecheck := true;
        initialize;
      end;
      write(dot);
      (*$I-*)
      reset (QT,'#5:queryfile.data');
      (*$I+*)
      if not filecheck then
      begin
        io := ioresult;
        if io <> 0 then
         begin
      filecheck := true;
      initialize;
    end;
  end;
  write(dot);
  buildmap;
  write(dot);
  filecheck := false;
end;


    procedure password;
       var
          n,z : integer;
          user : string;
          valid : boolean;


       begin {1}
         valid := false;
         clear;
         n := 1;
         lines(5);
         while (not valid) and (n <= 5) do
         begin {2}
           writeln('    PLEASE ENTER PASSWORD AND PRESS RETURN KEY');
           prompt;
           readln(keyboard,user);
           z := 1;
           while (not valid) and (z <= 5) do
           if userid[z] = user then
           begin{3}
             passwrd := true;
             valid := true
           end{3}
           else z := z + 1 ;
           if not valid then
             begin{4}
               n:= n + 1 ;
               lines(3);
               writeln('     ** PASSWORD INCORRECT **');
               writeln(chr(7));
               lines(1);
               if n > 5 then
               begin{5}
       writeln('         Please refer to the password instructions');
       writeln('         in the target information system handbook');
       writeln('         for the proper input.');
                 lines(4);
                 halt;
                 lines(4);
       writeln('        **  To restart system type R  **');
                 exit(password);
               end;{5}
```

```
                    end;{4}
                  end;{2}
               end;{1}



          procedure welcome1; forward;

          procedure welcome;

   begin
     clear;
     writeln(dots);
     writeln('        WELCOME TO THE TARGET INFORMATION SYSTEM');
     writeln(dots);
     lines(1);
     writeln('This program is a prototype target information system for');
     writeln('the Fire Support Coordination Center. It is designed to be');
     writeln('used by the personnel of the target information section of');
     writeln('the landing force FSCC in accordance with the principles');
     writeln('outlined in FMFM 7-1 (Fire Support Coordination).');
     lines(2);
     writeln('WARNING:');
     write(chr(14));
     writeln('      *** THIS FILE CONTAINS CONFIDENTIAL MATERIAL ***');
     write(chr(24));
     writeln(chr(29));
     welcome1;
  end;


        procedure welcome1;

      begin
  writeln(' The diskette file contains targets which are normally classifie
  writeln('confidential. The diskettte and all the backup copies should be
  writeln('handled as normal confidential documents and properly safeguarde
  writeln('Targets of a higher classification should not be entered on this
  writeln('file. Current emergency destruction procedures for confidential'
  writeln('material apply. Re-initializing the system removes all classifie
  writeln('information.');
       lines(1);
       spacebar;
     end;



        procedure welcome2;

      begin
        clear;
        lines(5);
```

```
iteln('    If at any time you become confused, in doubt about what');
iteln('to do next or what values to enter when you receive a prompt');
iteln('from the system ( ==> ), you can receive help or information');
iteln('by typing a ?.');
nes(4);
iteln('    If you need more information on how to operate the system,');
iteln('doctrinal guidelines for target information, security requirements');
iteln('or the types of formats used for target information, select option');
iteln('number 1 from the main command menu which follows.');
    lines(2);
    spacebar;
  end;



            procedure mminfo;

          begin
            clear;
            writeln(stars);
            lines(1);
            getfile('mminfo1.text');
            lines(1);
            spacebar;
            clear;
            lines(2);
            getfile('mminfo2.text');
            lines(1);
            spacebar;
            clear;
            lines(8);
            getfile('mminfo3.text');
            lines(5);
            spacebar;
          end;


            procedure mainmenu;

  begin
    clear;
    writeln(stars);
    writeln('      Target Information System Main Command Menu');
    writeln(dots);
    lines(1);
    writeln('     The options are:');
    lines(1);
    writeln('           1.  System Information');
    writeln('           2.  Work on Target File');
    writeln('           3.  Create a Special Target List');
    writeln('           4.  Perform Utility Functions');
    writeln('           5.  Initialize a New System');
    writeln('           6.  Information about this Menu');
    writeln('           7.  Halt Operation');
  lines(2);
end;




     begin (* interface*)

        getout := false;
        restart := false;
        menuloop := false;
        numbout := 0;
        passwrd := false;
        userid[1] := 'COULTER';
        userid[2] := 'E';
        userid[3] := 'e';
        userid[4] := 'MARINE';
        userid[5] := 'marine';
        menuchar := ['1','2','3','4','5','6','7','?'];
        password;
        if not passwrd then exit(interface);
        buildempty;
        openfiles;
        if getout then exit(interface);
        welcome;
        clear;
        welcome2;
        repeat
          mainmenu;
          select;
          read(ch);
          if ch in menuchar then
            begin
            if eoln (input) then readln;
            case ch of
               '1' : begin
                       loadmsg;
                       inform;
                     end;
               '2' : begin
                       loadmsg;
                       targetmod;
                     end;
               '3' : begin
                       loadmsg;
                       query;
                     end;
               '4' : begin
                       loadmsg;
                       utility;
                     end;
               '5' : begin
                       loadmsg;
```

```
                    initialize;
                    if restart then
                      begin
                        write('      Processing....');
                        buildmap;
                        lines(2);
                        writeln('                        FUNCTION  COMPLETE')
                        lines(1);
                        spacebar;
                      end;
                   end;
            '5','?' : mminfo;
            '7' : begin
                   getout := true;
                   halt;
                   clear;
                  end;
          end
         end
     else menuerror;
     until getout = true;
   end.


(*+++++++++++++++++++ END OF INTERFACE ++++++++++++++++++++++*)
```

```
          (*    GLOBALS.TEXT    *)

const

dot = ´.´;
stars = ´****************************************************************************************************
dots = ´................................................................................
spacebr = ´        PLEASE PRESS SPACEBAR TO CONTINUE´;
returner = ´      PLEASE PRESS RETURN TO CONTINUE´;
prompter = ´      ==>´;
selectop = ´      PLEASE ENTER OPTION NUMBER ´;
return = ´  Return to Previous Menu´;
numbervar = 19;
numbertgt = 300;



 type
     mvalue = set of char;
     state = (on,off);

      battledam = packed record
         DTGsurv : string[7];
         fireunit : string[6];
         ntrnds : string[10];
         damrep : char;
         damass : char;
         BDAtext : string[40]
      end;

      tarrec = packed record
           flag : packed array [1..numbervar] of state;
           tnum : string[6];
           loc : string[8];
           alt : string[4];
           desc : string[40];
           class : char;
           pri : char;
           stat : char;
           ttype : char;
           sa : char;
           rem : string[40];
           maprefer : string[20];
           sour : string[20];
           photonum : string[15];
           DTGact : string[7];
           photocord : string[8];
           acc : char;
           BDA : packed array [1..3] of battledam
        end;

         querytgt = packed record
```

```
          flag : state;
          ttype : char;
          class : char;
          sa : char;
          pri : char;
          acc : char;
          stat : char;
          tnum : string[6];
          loc : string[8];
          alt : string[4];
          desc : string[20]
        end;

    targetmap = packed array [0..numbertgt] of string[6];
    gridlocmap = packed array [0..numbertgt] of string[8];
    Qtarget = packed array [0..numbertgt] of querytgt;


var
      gridmap : gridlocmap;
      tgtmap : targetmap;
      target : file of record tgtrec : tarrec end;
      QT : file of record querrec : querytgt end;
      nochar, ch : char;
      restart, menuloop, passwrd, getout : boolean;
      userid : packed array [1 .. 5] of string;
      menuchar, menucar : mvalue;
      nostring, buffer, str : string;
      recnum, numbout, numbcheck, flg, range, flag : integer;
      filecheck, current, mandatoryitem, helpme, finished : boolean;
      endBDA, ok, trap, done, quit : boolean;
      x, i, ii, BDAcounter, n : integer;
      emptyTrec, currentgt : tarrec;
      emptyQrec, currentQT : querytgt;
      database : Qtarget;




(*................... SYSTEM GLOBALS ........................*)
```

```
                (*      QUERY .TEXT        *)


segment procedure query;

      var    charmenu : mvalue;
             tellused, first : boolean;
             amount, left, searcher, count, reccount : integer;
             index : char;
             Scheck, Pcheck, Acheck, statcheck : string[16];
             Tcheck, Ccheck, emptystring : string[16];
             cat : array [1..6] of string;



     procedure getdatabase;

     var  j : integer;

     begin
       clear;
       lines(4);
       writeln('   Data base being loaded into memory.....Please wait');
       lines(5);
       write('             Loading...');
       i := 0;
       j := 0;
       close(QT,lock);
       reset(QT,'#5:queryfile.data');
       while not eof(QT) do
       begin
         seek(QT,i);
         get(QT);
         database[i] := QT^.querrec;
         i := i + 1;
         j := j + 1;
         if j = 10 then
         begin
           write(dot);
           j := 0;
         end;
       end;
       lines(6);
       write('                         LOADING COMPLETE');
       reccount := i;
       for i := 1 to 4 do
         delay;
     end;
```

```
ocedure moresearch; forward;

ocedure searchdatabase;


egin
 for i := 0 to reccount - 1 do
 begin
 case searcher of
   1 : begin
        if first then
        begin
         if database[i].ttype = index then database[i].flag := on
        end
        else if (database[i].flag = on) and (database[i].ttype <> index)
        then database[i].flag := off;
      end;
   2 : begin
        if first then
        begin
         if database[i].class = index then database[i].flag := on
        end
        else if (database[i].flag = on) and (database[i].class <> index)
        then database[i].flag := off;
      end;
   3 : begin
        if first then
        begin
         if database[i].sa = index then database[i].flag := on
        end
        else if (database[i].flag = on) and  (database[i].sa <> index)
        then database[i].flag := off;
      end;
   4 : begin
        if first then
        begin
         if database[i].pri = index then database[i].flag := on
        end
        else if (database[i].flag = on) and (database[i].pri <> index)
        then database[i].flag := off;
      end;
   5 : begin
        if first then
        begin
         if database[i].acc = index then database[i].flag := on
        end
        else if (database[i].flag = on) and (database[i].acc <> index)
        then database[i].flag := off;
      end;
   6, 7 : moresearch;
   end;
   end;
 amount := 0;
 for i := 0 to reccount - 1 do
```

```
    if database[i].flag = on then amount := amount + 1;
  lines(2);
  writeln('          Number of targets in special list is ',amount);
  lines(2);
  spacebar;
    index := ' ';
    first := false;
  end;



      procedure moresearch;

begin
  case searcher of
    6 : begin {active}
          if first then
          begin
            if (database[i].stat = '1') or (database[i].stat = '2') or
            (database[i].stat = '3') or (database[i].stat = '4') then
            database[i].flag := on
          end
          else if (database[i].flag = on) and ((database[i].stat = '5')
            (database[i].stat = '6')) then database[i].flag := off;
        end;
    7 : begin {inactive}
          if first then
          begin
            if (database[i].stat = '5') or (database[i].stat = '6') the
            database[i].flag := on
          end
          else if (database[i].flag = on) and ((database[i].stat <> '5'
            (database[i].stat <> '6')) then database[i].flag := off;
       end;
     end;
   end;



        procedure DBtype;

        begin
          repeat
          clear;
          lines(2);
          getfile('typemenu.text');
          writeln('               R.',return);
          lines(1);
          select;
          read(ch);
          if ch in charmenu then
          begin
            if eoln(input) then readln;
            if ch in ['Q','q','R','r'] then exit(DBtype)
    else if ch = '?' then
      begin
        clear;
        getfile('tgttype.text');
        spacebar;
      end
    else begin
           searcher := 1;
           left := left - 1;
           index := ch;
           count := count + 1;
           case index of
             '1' : cat[count] := 'TANK';
             '2' : cat[count] := 'SEAD';
             '3' : cat[count] := 'INST';
             '4' : cat[count] := 'CBAT';
             '5' : cat[count] := ' OP ';
             '6' : cat[count] := 'TERR';
             '7' : cat[count] := 'VEH ';
             '8' : cat[count] := 'FORT';
             '9' : cat[count] := 'MISC';
           end;
           searchdatabase;
           exit(DBtype);
         end
  end
  else menuerror;
  until menuloop = true;
end;



procedure DBclass;

var  temp : string[2];

begin
  temp := '  ';
  repeat
    clear;
    lines(1);
    getfile('classmenu.text');
    writeln('             6.',return);
    lines(2);
    select;
    read(ch);
    if ch in charmenu then
    begin
      if eoln(input) then readln;
      if ch in ['6','Q','q','R','r'] then exit(DBclass)
      else if ch = '?' then
        begin
          clear;
          getfile('class.text');
```

```
            spacebar;
          end
        else begin
               searcher := 2;
               left := left - 1;
               count := count + 1;
               case ch of
                 '1' : index := 'A';
                 '2' : index := 'B';
                 '3' : index := 'C';
                 '4' : index := 'D';
                 '5' : index := 'E';
               end;
               case index of
                 'A' : temp := ' A';
                 'B' : temp := ' B';
                 'C' : temp := ' C';
                 'D' : temp := ' D';
                 'E' : temp := ' E';
               end;
               cat[count] := concat('Class',temp);
               searchdatabase;
               exit(DBclass);
             end
    end
    else menuerror;
    until menuloop = true;
  end;



procedure DBSAassgn;

begin
  repeat
  clear;
  lines(2);
  getfile('samemu.text');
  writeln('                    R.',return);
  lines(1);
  select;
  read(ch);
  if ch in charmenu then
  begin
    if eoln(input) then readln;
    if ch in ['R','r','Q','q'] then exit(DBSAassgn)
    else if ch = '?' then
      begin
        clear;
        getfile('sa.text');
        spacebar;
      end
    else begin
           searcher := 3;
```

```
            left := left - 1;
            index := ch;
            count := count - 1;
            case index of
               '1' : cat[count] := 'ARTY';
               '2' : cat[count] := 'NGF ';
               '3' : cat[count] := 'AIR ';
               '4','5','6','7' : cat[count] := 'COMB';
               '8' : cat[count] := 'OTHR';
               '9' : cat[count] := 'NONE';
            end;
            searchdatabase;
            exit(DBSAassgn);
        end
  end
  else menuerror;
  until menuloop = true;
end;



procedure DBpri;

var temp : string[3];

begin
  repeat
  clear;
  lines(1);
  getfile('tgtprimenu.text');
  writeln('                  5.',return);
  lines(1);
  select;
  read(ch);
  if ch in charmenu then
  begin
    if eoln(input) then readln;
    if ch in ['5','R','r','Q','q'] then exit(DBpri)
    else if ch = '?' then
      begin
      clear;
      getfile('priority.text');
      spacebar;
    end
    else begin
            searcher := 4;
            index := ch;
            left := left - 1;
            count := count + 1;
            case index of
               '1' : temp := 'I  ';
               '2' : temp := 'II ';
               '3' : temp := 'III';
               '4' : temp := 'IV ';
            end;
            cat[count] := concat('Pri ',temp);
            searchdatabase;
            exit(DBpri);
          end
  end
  else menuerror;
  until menuloop = true;
end;



procedure DBacc;

begin
  repeat
  clear;
  lines(2);
  getfile('tgtaccmenu.text');
  writeln('                 5.',return);
  lines(1);
  select;
  read(ch);
  if ch in charmenu then
  begin
    if eoln(input) then readln;
    if ch in ['5','R','r','q','Q'] then exit(DBacc)
    else if ch = '?' then
      begin
        clear;
        getfile('tgtacc.text');
        spacebar;
      end
    else begin
            searcher := 5;
            index := ch;
            left := left - 1;
            count := count + 1;
            case index of
              '1' : cat[count] := 'CONFIRMED';
              '2' : cat[count] := 'PROBABLE ';
              '3' : cat[count] := 'POSSIBLE ';
              '4' : cat[count] := 'UNKNOWN  ';
            end;
            searchdatabase;
            exit(DBacc);
          end
  end
  else menuerror;
  until menuloop = true;
end;
```

```
 procedure DBstatus;

 var  act : string[8];
      loop : boolean;

begin
 act := '        ';
 menucar := ['1','2','?'];
 loop := false;
 repeat
   clear;
   lines(5);
   writeln('      ENTER TARGET STATUS--ACTIVITY');
   lines(1);
   writeln('       The options are:');
   lines(1);
   writeln('          1.  Active');
   writeln('          2.  Inactive');
   lines(1);
   writeln('        PLEASE ENTER OPTION NUMBER AND PRESS RETURN';
   prompt;
   read(ch);
   if ch in menucar then
   begin
     if not eoln(input) then readln;
     if ch = '1' then
     begin
       searcher := 6;
       searchdatabase;
       loop := true;
       act := ' ACTIVE ';
     end;
     if ch = '2' then
     begin
       searcher := 7;
       searchdatabase;
       loop := true;
       act := 'INACTIVE';
     end;
     if ch = '?' then
     begin
       lines(1);
riteln('    An Active target is one which is found in the Target list or');
riteln('the list of targets. An inactive target is in the deadfile.');
       lines(1);
       spacebar;
     end;
     end
     else if (ch in ['Q','q']) or (eoln(input)) then exit(DBstatus)
     else menuerror;
   until loop = true;
 left := left - 1;
```

```
    count := count + 1;
    cat[count] := act;
  end;



       procedure catmenu;

    begin
      writeln('             Categories for Special Listing');
      writeln(dots);
      lines(1);
      writeln('  The listing can contain ',left,' items from the below me
      lines(1);
      writeln('              1.  Target type                   ',Tcheck);
      writeln('              2.  Classification                ',Ccheck);
      writeln('              3.  Supporting arm assigned       ',Scheck);
      writeln('              4.  Priority                      ',Pcheck);
      writeln('              5.  Accuracy                      ',Acheck);
      writeln('              6.  Status                        ',statcheck);
      writeln('            * P.  Process information');
      lines(1);
      writeln('        Special list currently contains ',amount,' targets.
      if amount <= 0 then writeln('                  Please start a new li
      lines(1);
    end;



       procedure catproc;

       var  taken : string[16];

       begin
         taken := 'Already Selected';
         repeat
           if count >= 6 then
           begin
             clear;
             lines(4);
             writeln('     No more categories available for special list.
             writeln('            Please print target listing');
             lines(2);
             spacebar;
             exit(catproc);
           end;
           clear;
           case searcher of
             0 : ;
             1 : Tcheck := taken;
             2 : Ccheck := taken;
             3 : Scheck := taken;
             4 : Pcheck := taken;
      5 : Acheck := taken;
      6, 7 : statcheck := taken;
    end;
    catmenu;
    select;
    read(ch);
    if ch in charmenu then
    begin
      if eoln(input) then readln;
      case ch of
        '1' : begin
                if Tcheck = taken then tellused := true
                else DBtype;
              end;
        '2' : begin
                if Ccheck = taken then tellused := true
                else DBclass;
              end;
        '3' : begin
                if Scheck = taken then tellused := true
                else DBSAassgn;
              end;
        '4' : begin
                if Pcheck = taken then tellused := true
                else DBpri;
             end;
        '5' : begin
                if Acheck = taken then tellused := true
                else DBacc;
              end;
        '6' : begin
                if statcheck = taken then tellused := true
                else DBstatus;
              end;
        'P','p','R','r','Q','q' : exit(catproc);
        '7','8','9' : menuerror;
        '?' : begin
                lines(1);
                writeln('     See prior menu for information');
                lines(1);
                spacebar;
               end;
      end
    end
    else menuerror;
  if tellused then
    begin
      clear;
      lines(5);
      writeln('     Category has already been selected. Please');
      writeln('     choose another category from the unused items');
      writeln('     on the menu listing. To start a new listing,');
      writeln('     choose option 7 to return to the main menu.');
      lines(3);
```

```
          spacebar;
          tellused := false;
        end;
      until menuloop = true;
    end;



procedure screenlist;

    var   star : char;
          pribuff : string[3];
          sabuff : string[4];
          holdcar : char;
          pager : integer;



  procedure header;

        var  listing : string;

        begin
          listing := ' ';
          lines(1);
          writeln('                        SPECIAL TARGET LISTING');
          writeln('                        ----------------------');
          write('Categories:');
          if count > 6 then count := 6;
          for i := 1 to count do
            listing := concat(listing,' ',cat[i]);
          writeln(listing);
          lines(1);
          writeln('TGT NO   CL PRI  LOCATION   ALT  SAASG  DESCRIPTION
          writeln('------   -- ---  --------   ---- -----  -----------
        end;


   begin{screenlist}
     clear;
     header;
     pribuff := '   ';
     sabuff := '    ';
     pager := 8;
     for i := 0 to reccount - 1 do
     begin
       star := ' ';
       if database[i].flag = on then
       begin
         holdcar := database[i].pri;
         case holdcar of
           '1' : pribuff := 'I  ';
           '2' : pribuff := 'II ';
           '3' : pribuff := 'III';
          '4' : pribuff := 'IV ';
        end;
        holdcar := database[i].sa;
        case holdcar of
          '1' : sabuff := 'ARTY';
          '2' : sabuff := 'NGF ';
          '3' : sabuff := 'AIR ';
          '4','5','6','7' : sabuff := 'COMB';
          '8' : sabuff := 'OTHR';
          '9' : sabuff := 'NONE';
        end;
f (database[i].stat = '1') or (database[i].stat = '2') then star := '*';
rite(database[i].tnum,star,'  ',database[i].class,'  ',pribuff,'  ');
rite(database[i].loc,'   ',database[i].alt:4,'  ',sabuff,'   ');
riteln(database[i].desc);
      pager := pager + 1;
      if pager = 22 then
      begin
        lines(1);
        spacebar;
        clear;
        lines(2);
        pager := 0;
      end;
    end;
  end;
  lines(1);
  writeln('    NOTE:  * indicates target list');
  lines(1);
  spacebar;
end;


      procedure reset;

      begin
        tellused := false;
        amount := 0;
        left := 6;
        count := 0;
        searcher := 0;
        first := true;
        Tcheck := emptystring;
        Ccheck := emptystring;
        Pcheck := emptystring;
        Acheck := emptystring;
        Scheck := emptystring;
        statcheck := emptystring;
        for i := 0 to reccount - 1  do
          database[i].flag := off;
        for i := 1 to 6 do
            cat[i] := emptystring;
      end;
```

```
procedure queryproc;

  begin
    count := 0;
    repeat
    clear;
    lines(2);
    writeln('              SPECIAL TARGET LISTINGS');
    writeln(dots);
    lines(2);
    writeln('       The options are:');
    lines(1);
    writeln('              1.  Form a special target listing');
    writeln('              2.  Continue to process');
    writeln('              3.  Write the special list to the screen');
    writeln('              4.  Information about this procedure');
    writeln('              5.',return);
    lines(1);
    select;
    read(ch);
    if ch in ['1','2','3','4','5','R','r','?'] then
    begin
      if eoln(input) then readln;
      case ch of
        '1' : begin
                reset;
                catproc;
              end;
        '2' : catproc;
        '3' : screenlist;
        '4','?' : begin
                    clear;
                    getfile('queryinfo.text');
                    spacebar;
                  end;
        '5','R','r' : exit(query);
      end
    end
    else menuerror;
    until menuloop = true;
  end;




      begin {query}
        reccount := 0;
        emptystring := '                    ';
        searcher := 0;
charmenu := ['1','2','3','4','5','6','7','8','9','R','r','Q','q','P','
        getdatabase;
        reset;
        queryproc;
```

```
end;

(*..................END  QUERY........................*)
```

```
(*      BDA.TEXT     *)

     segment procedure targetmod;

     var
        stat1 : string[6];
        stat2 : string[3];
        ttypebuf : string[4];
        pribuf : string[3];
        sabuf : string[14];
        accbuf : string[9];
        duplicate, fetchback, outprocess, first : boolean;


     procedure fetchtgt(grid : integer); forward;
     procedure readin; forward;
     procedure checkDTG(var strng : string; var check:boolean); forward;
     procedure process; forward;
     procedure cutstring(strngsize : integer); forward;
     procedure putinfile; forward;



segment procedure newBDA;


procedure DTGofBDA;


begin
  currentgt.BDA[BDAcounter].DTGsurv := '         ';
  while not finished do
    begin
      clear;
      lines(6);
      writeln('      ENTER DTG TARGET WAS ATTACKED...6 digits and 1 lette
      prompt;
      readin;
      checkDTG(str,ok);
      if quit then exit(DTGofBDA);
      if helpme or not ok then
        begin
          finished := false;
          lines(1);
          getfile('dtgofbda.text');
          lines(1);
          returnbar;
        end;
        if  (flg = 2) and finished  then
        currentgt.BDA[BDAcounter].DTGsurv := str;
      end;
    end;
    procedure firingunit;

egin
 currenttgt.BDA[BDAcounter].fireunit := '      ';
 while not finished do
  begin
   lines(2);
   writeln('     ENTER FIRING UNIT....do not exceed 6 characters');
   prompt;
   readin;
   if helpme then
   begin
     finished := false;
     lines(1);
     getfile('funit1.text');
     lines(1);
     returnbar;
   end
   else if length(str) > 6 then
     begin
       lines(1);
       getfile('funit2.text');
       lines(1);
       finished := false;
       currenttgt.flag[n] := on;
       returnbar;
     end
   else if quit then exit(firingunit)
   else if (flg = 2) and finished  then
   currenttgt.BDA[BDAcounter].fireunit := str;
 end;
nd;


    procedure rounds;

egin
 currenttgt.BDA[BDAcounter].ntrnds := '            ';
 while not finished do
   begin
   lines(2);
   writeln('     ENTER NUMBER AND TYPE OF ROUNDS FIRED');
   prompt;
   readin;
   if helpme then
   begin
     finished := false;
     lines(1);
     getfile('rounds.text');
     lines(1);
     returnbar;
   end
   else if length(str) > 12 then
     begin
```

```
         lines(1);
         getfile('rounds1.text');
         lines(1);
         finished := false;
         currentgt.flag[n] := on;
         returnbar;
      end
      else if quit then exit(rounds)
      else if (flg = 2) and finished then
      currentgt.BDA[BDAcounter].ntrnds := str;
     end;
end;


     procedure damagemenu( param : integer);

     var  kind : string[8];

     begin
       if param = 1 then kind := 'REPORTED'
       else kind := 'ASSESSED';
       clear;
       lines(5);
       writeln('      ENTER DAMAGE ',kind);
       lines(1);
       getfile('damagemen.text');
       lines(2);
     end;



     procedure damagrept;

   begin
       currentgt.BDA[BDAcounter].damrep := '9';
       repeat
         damagemenu(1);
         select;
         read(ch);
         if ch in ['1'..'8'] then
         begin
           if eoln(input) then readln;
           finished := true;
           currentgt.BDA[BDAcount].damrep := ch;
         end
         else if eoln(input) then exit(damagrept)
         else if ch in ['Q','q'] then
         begin
           quit := true;
           exit(damagrept);
         end
         else if ch = '?' then
         begin
           lines(1);
```

```
          getfile('damrep.text');
          lines (1);
          returnbar;
        end
      else menuerror;
     until finished = true;
    end;


     procedure damagassd;

     begin
       currentgt.BDA[BDAcounter].damass := '9';
       repeat
         damagemenu(2);
         select;
         read(ch);
         if ch in ['1'..'8'] then
         begin
           if eoln(input) then readln;
           finished := true;
           currentgt.BDA[BDAcount].damass := ch;
         end
         else if eoln(input) then exit(damagassd)
         else if ch in ['Q','q'] then
         begin
           quit := true;
           exit(damagassd);
         end
         else if ch = '?' then
         begin
           lines(1);
           getfile('damass.text');
           lines(1);
           returnbar;
       end
     else menuerror;
    until finished = true;
    end;



procedure BDAremarks;
var  x : integer;

begin
  currentgt.BDA[BDAcounter].BDAtext := nostring;
  while not finished do
  begin
    clear;
    lines(6);
    writeln('      ENTER BDA....do not exceed one line');
    prompt;
    readln;
    if helpme then
    begin
      finished := false;
      lines(1);
      getfile('bdarem.text');
      lines(1);
      returnbar;
    end;
    if quit then exit(BDAremarks);
    if  (flg = 2) and finished then
     begin
       buffer := '                                            ';
       cutstring(42);
       currenttgt.BDA[BDAcount].BDAtext := str;
     end;
  end;
end;




      procedure BDAinfo;

      begin
        clear;
        lines(2);
        writeln('                 BATTLE DAMAGE ASSESSMENT');
        lines(2);
        writeln('      For information on adding a target surveillance'
        writeln('              to the target file, type a ?.');
        lines(3);
        writeln('       **   To continue, press the RETURN key.   **')
        prompt;
        read(ch);
        if ch = '?' then
          begin
            clear;
            lines(2);
            getfile('bdainfo.text');
            lines(3);
            spacebar;
            clear;
          end;
       end;




      begin {newBDA}
        if not current then
        begin
          BDAinfo;
          fetchtgt(1);
      if quit then exit(newBDA);
      i := 1;
      while (currenttgt.flag[16 + i] = off) do
        begin
          i := i + 1;
          if i = 4 then
          begin
            i := 1;
            currenttgt.flag[16 + i] := on;
          end;
        end;
      BDAcounter := i;
      n := 16 + i;
    end;
    endBDA := false;
    while not endBDA do
    begin
      for ii := 1 to 6 do
      begin
          finished := false;
          case ii of
            1 : DTGofBDA;
            2 : firingunit;
            3 : rounds;
            4 : damagrept;
            5 : damagassd;
            6 : begin
                  BDAremarks;
                  endBDA := true;
                end;
           end;
       end;
     end;
     if current then exit(newBDA)
     else begin
         with currenttgt do
            begin
              if stat = '2' then stat := '1'
              else if stat = '4' then stat := '3'
              else if stat = '6' then stat := '5'
            end;
         with currentCT do
            begin
              if stat = '2' then stat := '1'
              else if stat = '4' then stat := '3'
              else if stat = '6' then stat := '5'
            end;
         putinfile;
      end;
end;
```

```
          (*       TGTPROCS.TEXT    *)


(*................................................................*)


   procedure mandmsg;

   begin
     lines(1);
     getfile('mandmsg.text');
     lines(1);
     spacebar;
   end;


   procedure readin;

   var  len : integer;

   begin
   helpme := false;
   readln(str);
   len := length(str);
   if len = Ø then flg := Ø
   else if str[1] = '?' then flg := 1                        {help}
   else if (len = 1) and (str[1] in ['Q','q']) then flg := 3 {quit}
   else flg := 2;                                            {continu
   case flg of
     Ø : begin
         if mandatoryitem then mandmsg
         else finished := true;
         end;
     1 : helpme := true;
     2 : begin
           currenttgt.flag[n] := off;
           finished := true;
         end;
     3 : quit := true;
       end
     end;



     procedure checknum(var strng : string ; var check : boolean);

     var x,i : integer;

     begin
       check := true;
```

```
        x := length(strng);
        if x = Ø then
          begin
            fetchback := true;
            exit(checknum);
          end;
        if x <> 6 then
        begin
          check := false;
          exit(checknum);
        end;
        for i := 1 to 2 do
        if not (strng[i] in ['A'..'Z']) then
          begin
            check := false;
            writeln('    Use upper case letters for target designator');
            exit(checknum);
          end;
        for i := 3 to 6 do
          if not (strng[i] in ['Ø'..'9']) then check := false;
        end;



procedure checkdigit (var strng:string; var check : boolean; rng : integer);

var  i, x : integer;

begin
  check := true;
  x := length(strng);
  if x = Ø then
          begin
            fetchback := true;
            exit(checkdigit);
          end;
  if x <> rng then
  begin
    check := false;
    exit(checkdigit);
  end;
  for i := 1 to rng do
    if not (strng[i] in ['Ø'..'9']) then check := false;
end;


procedure cutstring ;

(*   (strngsize : integer) removed for fwd dec    *)
```

```
var    cutter : integer;

begin
  if length(str) > strngsize then
  begin
    for cutter := 1 to strngsize do
      buffer[cutter] := str[cutter];
    str := buffer;
  end;
end;




   procedure checkDTG ;


   (*     (var strng : string; var check : boolean) removed for fwd dec

   var    i : integer;

      begin

     check := true;
     if length(str) = 0 then exit(checkDTG);
     if length(strng) <> 7 then
     begin
       check := false;
       exit(checkDTG);
     end;
     for i := 1 to 6 do
       if not(strng[i] in ['0'..'9']) then
       begin
         check := false;
         exit(checkDTG);
       end;
     if not (strng[7] in ['A'..'Z']) and not (strng[7] in ['a'..'z'])
       check := false;
   end;

  (*..................................................................*)
  (*   ADDTARGET.TEXT      *)


procedure verifynum;

begin
  for i := 1 to numbertgt do
    if tgtmap[i] = str then
    begin
      lines(2);
      writeln('  **   Target number already exists in target file    **');
      lines(1);
      writeln('      Please use a different target number. To reuse');
      writeln('       this number, you must delete the target using');
      writeln('                   option 6 of the menu.');
      lines(1);
      spacebar;
      duplicate := true;
      done := true;
    end;
end;


procedure tgtnum;

begin
  while not finished do
  begin
    clear;
    lines(6);
    writeln('       ENTER TARGET NUMBER');
    prompt;
    readin;
    if quit then exit (tgtnum);
    checknum(str,ok);
    if helpme or not ok then
    begin
      finished := false;
      lines(1);
      getfile('tnum.text');
      lines(1);
      returnbar;
    end;
    if (flg = 2) and finished then
    begin
      duplicate := false;
      verifynum;
      if duplicate then
        begin
          quit := true;
          exit(tgtnum);
        end;
```

```
            currenttgt.tnum := str;
            currentCT.tnum := str;
          end;
        end;
      end;



procedure tgtloc;

begin
  range := 6;
  while not finished do
    begin
      clear;
      lines(5);
      writeln('      ENTER TARGET LOCATION....use 6 digits');
      prompt;
      readln;
      checkdigit(str,ok,range);
      if quit then exit(tgtloc);
      if helpme or not ok then
        begin
          finished := false;
          lines(1);
          getfile('tgtloc.text');
          lines(1);
          returnbar;
        end;
        if (flg = 2) and finished then
          begin
            currenttgt.loc := str;
            currentCT.loc := str;
          end;
      end;
    end;



procedure tgtdesc;

begin
  while not finished do
  begin
    clear;
    lines(5);
    writeln('      ENTER TARGET DESCRIPTION....do not exceed one line');
    prompt;
    readln;
    if quit then exit(tgtdesc);
    if helpme then
  begin
    finished := false;
    lines(1);
    getfile('tgtdesc.text');
    lines(1);
    returncar;
  end;
  if (flg = 2) and finished then
  begin
    buffer := '                                                    ';
    cutstring(26);
    currenttgt.desc := str;
    buffer := '                              ';
    cutstring(20);
    currentWT.desc := str;
  end;
end;
d;
```

```
ocedure tgtclass;

gin
repeat
clear;
lines(3);
getfile('classmenu.text');
lines(1);
select;
read(ch);
if ch in menuchar then
begin
  if eoln (input) then readln;
    currenttgt.flag[n] := off;
    finished := true;
  case ch of
   '1' : begin
             currenttgt.class := 'A';
             currentWT.class := 'A';
         end;
   '2' : begin
             currenttgt.class := 'B';
             currentWT.class := 'B';
         end;
   '3' : begin
             currenttgt.class := 'C';
             currentWT.class := 'C';
```

```
          end;
    '4' : begin
            currenttgt.class := 'D';
            currentQT.class := 'D';
          end;
    '5' : begin
            currenttgt.class := 'E';
            currentQT.class := 'E';
          end;
    '6','7','8' : begin
                    currenttgt.flag[n] := ch;
                    finished := false;
                    menuerror;
                  end;
    '?' : begin
            clear;
            lines(1);
            getfile('class.text');
            lines(1);
            spacebar;
            currenttgt.flag[n] := ch;
            finished := false;
          end
      end
  end
  else if ch in ['Q','q'] then
          begin
            quit := true;
            exit(tgtclass);
          end
  else if (eoln(input)) then menumsg
  else menuerror;
  until finished = true;
end;



procedure tgtpri;

begin
  repeat
  clear;
  lines(4);
  getfile('tgtprimenu.text');
  lines(1);
  select;
  read(ch);
  if ch in menuchar then
  begin
    if eoln (input) then readln;
    if ch in ['1'..'4'] then
    begin
      currenttgt.flag[n] := off;
      finished := true;
      currenttgt.pri := ch;
      currentwT.pri := ch;
    end;
  if ch = '?' then
     begin
       clear;
       lines(1);
       getfile('priority.text',;
       lines(3);
       spacebar;
     end;
  if ch in ['5','S','R','r'] then menuerror;
  end
  else if ch in ['Q','q'] then
           begin
             quit := true;
             exit(tgtpri);
           end
  else if (eoln(input)) then menumsg
  else menuerror;
  until finished = true;
end;


  procedure tgtstatus;

  var loop : boolean;
  code : integer;


  procedure active;

  begin
    loop := false;
    repeat
      clear;
      lines(5);
      writeln('      ENTER TARGET STATUS--ACTIVITY',;
      lines(1);
      writeln('       The options are:');
      lines(1);
      writeln('           1.  Active');
      writeln('           2.  Inactive');
      lines(1);
      writeln('      PLEASE ENTER OPTION NUMBER AND PRESS RETURN');
      prompt;
      read(ch);
      if ch in menuchar then
      begin
        if not eoln(input) then readln;
        if ch = '1' then
        begin
          code := 0;
```

```
            loop := true;
          end;
          if ch = '2' then
          begin
            code := 12;
            loop := true;
          end;
          if ch = '?' then
          begin
            lines(1);
writeln('     An active target is one which is found in the target list o
writeln('the list of targets. An inactive target is in the deadfile.');
            lines(1);
            spacebar;
          end;
          if ch in ['3','4','5','6','R','r'] then menuerror;
          end
          else if ch in ['Q','q'] then
          begin
            quit := true;
            exit(active);
          end
          else if (eoln(input)) then menurse
          else menuerror;
        until loop = true;
    end;

    procedure listed;

    begin
      loop := false;
      repeat
        clear;
        lines(5);
        writeln('       ENTER TARGET STATUS--HOW LISTED');
        lines(1);
        writeln('        The options are:');
        lines(1);
        writeln('            1.  Target List',;
        writeln('            2.  List of targets');
        lines(1);
        select;
        read(ch);
        if ch in menuchar then
        begin
          if eoln(input) then readln;
          if ch = '1' then
          begin
            code := code + 3;
            loop := true;
          end;
          if ch = '2' then
          begin
            code := code + 6;
          loop := true;
        end;
        if ch = '?' then
        begin
          lines(2);
writeln('The Target List refers to the Target List of the highest');
writeln('headquarters. If a target is not on the Target List then it');
writeln('             is on the list of targets.');
          lines(1);
          spacebar;
        end;
        if ch in ['3','4','5','6','7','8'] then menuerror;
        end
        else if ch in ['Q','q'] then
             begin
               quit := true;
               exit(listed);
             end
    else if (eoln(input)) then mainmsg
    else menuerror;
  until loop = true;
  end;

  procedure attacked;

  begin
  loop := false;
  repeat
      clear;
      lines(5);
      writeln('      ENTER TARGET STATUS--SUPPORTING ARMS ATTACK');
      lines(1);
      writeln('       The options are:');
      lines(1);
      writeln('            1.  Attacked');
      writeln('            2.  Not attacked');
      lines(1);
      select;
      read(ch);
      if ch in menuchar then
      begin
        if eoln(input) then readln;
        if ch = '1' then
        begin
          code := code + 7;
          loop := true;
        end;
        if ch = '2' then
        begin
          code := code + 8;
          loop := true;
        end;
        if ch = '?' then
        begin
```

```
              lines(2);
writeln('Attacked targets are those attacked by supporting arms');
writeln('for which there is a surveillance or damage reported.');
              lines(1);
              spacebar;
            end;
            if ch in ['3','4','5','6','a','r'] then menuerror;
            end
            else if ch in ['Q','q'] then
                begin
                  quit := true;
                  exit(attacked);
                end
            else if (eoln(input)) then badmsg
            else menuerror;
        until loop = true;
     end;


     begin          {tgtstatus}
       repeat
       lines(2);
       active;
       if quit then exit(tgtstatus);
       listed;
       if quit then exit(tgtstatus);
       attacked;
       if quit then exit(tgtstatus);
       finished := true;
       case code of
        14 : ch := '1';
        12 : ch := '2';
        13 : ch := '3';
        16 : ch := '4';
   22, 24 : begin
              lines(2);
              writeln('     Combination not possible...please reenter sta
              spacebar;
              finished := false;
             end;
        25 : ch := '5';
        27 : ch := '6';
        end;
        until finished;
        if ch in ['1','3','5'] then numtcheck := numtcheck - 1;
        currenttgt.flag[n] := off;
        currenttgt.stat := ch;
        currentqT.stat := ch;
      end;
```

```
rocedure tgttype;
ar    menunum : mvalue;

egin
 menunum := ['1','2','3','4','5','6','7','8','9'];
 repeat
     clear;
     lines(2);
     getfile('typemenu.text');
     lines(1);
     select;
     read(ch);
     if ch in menunum then
     begin
       if eoln(input) then readln;
       finished := true;
     end
     else if ch in ['Q','q'] then
     begin
       quit := true;
       exit(tgttype);
     end
     else if (eoln(input)) then
     begin
       clear;
       lines(5);
       mandmsg;
     end
     else if ch = '?' then
     begin
       lines(1);
       getfile('tgttype.text');
       lines(1);
       returncar;
     end
     else menuerror;
   until finished;
   currenttgt.ttype := ch;
   currenttgt.flag[n] := off;
   currentGT.ttype := ch;
nd;



rocedure tgtalt;

     var x, y : integer;


egin
 currenttgt.alt := nostring;
 currentGT.alt := nostring;
 while not finished do
  begin
```

```
    clear;
    lines(8);
    writeln('      ENTER TARGET ALTITUDE--use meters only');
    prompt;
    readln;
    if quit then exit(tgtalt);
    x := length(str);
    for y := 1 to x do
      if not (str[y] in ['0'..'9']) then helpme := true;
    if x > 5 then helpme := true;
    if helpme then
    begin
      finished := false;
      lines(1);
      getfile('tgtalt.text');
      lines(1);
      returncar;
    end;
    if (flg = 2) and finished then
    begin
      currenttgt.alt := str;
      currentT.alt := str;
    end;
  end;
end;




procedure SAassgn;

var   menunum : mvalue;

begin
  menunum := ['1','2','3','4','5','6','7','8','9'];
    repeat
      clear;
      lines(2);
      getfile('samenu.text');
      lines(1);
      select;
      read(ch);
      if ch in menunum then
      begin
        if eoln(input) then readln;
        finished := true;
      end
      else if ch in ['Q','q'] then
      begin
        quit := true;
        exit(SAassgn);
      end
      else if ch = '?' then
        begin
          clear;
          lines(2);
          getfile('sa.text');
          lines(1);
          returncar;
        end
        else if eoln(input) then
        begin
          currenttgt.sa := '9';
          currentQT.sa := '9';
          exit(SAassgn);
        end
        else menuerror;
    until finished;
    currenttgt.sa := ch;
    currenttgt.flag[n] := off;
    currentQT.sa := ch;
end;



procedure remarks;

begin
  currenttgt.rem := nostring;
  while not finished do
  begin
    clear;
    lines(6);
    writeln('      ENTER REMARKS CONCERNING TARGET....do not exceed one line'
    prompt;
    readin;
    if helpme then
    begin
      finished := false;
      lines(1);
      getfile('remarks.text');
      lines(1);
      returncar;
    end;
    if quit then exit(remarks);
    if (flg = 2) and finished then
    begin
      buffer := '                                        ';
      cutstring(40);
      currenttgt.rem := str;
    end;
  end;
end;
```

```
procedure mapref;

begin
  currentgt.maprefer := nostring;
  while not finished do
  begin
    clear;
    lines(6);
    writeln('      ENTER TARGET MAP REFERENCE....do not exceed 20 charact
    prompt;
    readin;
    if helpme then
    begin
      finished := false;
      lines(1);
      getfile('mapref.text');
      lines(1);
      returnbar;
    end;
    if quit then exit(mapref);
    if (flg = 2) and finished then
    begin
      buffer := '                     ';
      cutstring(20);
      currentgt.maprefer := str;
    end;
  end;
end;



procedure source;

begin
  currentgt.sour := nostring;
  while not finished do
  begin
    clear;
    lines(6);
    writeln('      ENTER SOURCE OF TARGET....do not exceed 20 characters'
    prompt;
    readin;
    if helpme then
    begin
      finished := false;
      lines(1);
      getfile('sour.text');
      lines(1);
      returnbar;
    end;
    if quit then exit(source);
    if (flg = 2) and finished then
    begin
      buffer := '                     ';
```

```
     cutstring(20);
     currenttgt.sour := str;
   end;
 end;
nd;



rocedure afotonum;

egin
 currenttgt.photonum := nostring;
 while not finished do
 begin
   clear;
   lines(6);
   writeln('      ENTER AERIAL PHOTO NUMBER');
   prompt;
   readin;
   if helpme then
   begin
     finished := false;
     lines(1);
     getfile('afotonum.text');
     lines(1);
     returnbar;
   end;
   if quit then exit(afotonum);
   if (flg = 2) and finished then
   begin
     buffer := '                ';
     cutstring(15);
     currenttgt.photonum := str;
   end;
 end;
nd;

       procedure photogrid;

       begin
range := 5;
while not finished do
begin
  clear;
  lines(6);
  getfile('photogrid1.text');
  prompt;
  readin;
  if flg = 0 then
  begin
    currenttgt.photocord := nostring;
    exit(photogrid);
  end;
  if (flg = 2) and (length(str) = 1) and (str[1] in ['S','s']) then
```

```
      begin
        currentgt.photocord := currentgt.loc;
        exit(photogrid);
      end;
    checkdigit (str,ok,range);
    if quit then exit (photogrid);
    if helpme or not ok then
    begin
      finished := false;
      lines(1);
      getfile('photogrid2.text');
      lines(1);
      returnbar;
    end;
    if (flg = 2) and finished then currentgt.photocord := str;
  end;
end;


        procedure DTGactive;

begin
  currentgt.DTGact := nostring;
  while not finished do
    begin
      clear;
      lines(6);
      writeln('      ENTER DTG TARGET WAS ACTIVATED...6 digits and 1 lett
      prompt;
      readln;
      check TG(str,ok);
      if quit then exit(DTGactive);
      if helpme or not ok then
        begin
          finished := false;
          lines(1);
          getfile('dtgact.text');
          lines(1);
          returnbar;
        end;
        if (flg = 2) and finished then currentgt.DTGact := str;
      end;
    end;



  procedure tgtaccuracy;

  begin
    currentgt.acc := '4';
    currentGT.acc := '4';
    repeat
    clear;
  lines(4);
  getfile('tgtaccmenu.text');
  lines(1);
  select;
  read(ch);
  if ch in menuchar then
  begin
    if eoln (input) then readln;
    if ch in ['1'..'4'] then finished := true;
    if ch in ['5','6','R','r'] then menuerror;
    if ch = '?' then
    begin
      lines(1);
      getfile('tgtacc.text');
      lines(1);
      returnbar;
    end;
   end
   else if ch in ['Q','q'] then
        begin
          quit := true;
          exit(tgtaccuracy);
        end
  else if eoln(input) then exit(tgtaccuracy)
  else menuerror;
  until finished = true;
      currenttgt.flag[n] := off;
      currenttgt.acc := ch;
      currentCT.acc := ch;
end;
```

```
              (*       CHANGE PROCEDURE        *)

    procedure caseproc; forward;
    procedure displaytgt; forward;


 procedure change;

 var  punchout, yes : boolean;


 procedure changeinfo;

 begin
   clear;
   lines(2);
   getfile('changeinfo.text');
   lines(2);
   spacebar;
   clear;
 end;


              procedure yescheck;

    begin
      finished := false;
      yes := false;
      lines(2);
      writeln('     Change ?  Y(es)  N(o)');
      prompt;
      read(ch);
      if (ch = 'Y') or (ch = 'y') then yes := true
      else if ( ch = 'Q') or ( ch = 'q') then punchout := true;
    end;



    procedure cngproc2;

begin
  with currenttgt do
    begin
      clear;
      lines(8);
      writeln('  Current air photo grid is...',photocord);
      yescheck;
      if yes then photogrid;
      if punchout then exit(cngproc2);
      clear;
      lines(8);
      writeln('  Current DTG target activated is...',DTGact);
      yescheck;
      if yes then DTGactive;
      if punchout then exit(chgproc2);
      clear;
      lines(6);
      writeln('  Current accuracy is...',acccur);
      yescheck;
      if yes then tgtaccuracy;
      if punchout then exit(chgproc2);
    end;
end;


procedure changeproc;

begin
  quit := false;
  caseproc;
    with currenttgt do
      begin
        clear;
        lines(2);
        writeln('         TARGET  NUMBER  ',tnum);
        lines(6);
        writeln('  Current location is...',loc);
        yescheck;
        if yes then tgtloc;
        if punchout then exit(changeproc);
        clear;
        lines(6);
        writeln('  Current description is...',desc);
        yescheck;
        if yes then tgtdesc;
        if punchout then exit(changeproc);
        clear;
        lines(5);
        writeln('  Current class is...',class);
        yescheck;
        if yes then tgtclass;
        if punchout then exit(changeproc);
        clear;
        lines(5);
        writeln('  Current priority is...',pricur);
        yescheck;
        if yes then tgtpri;
        if punchout then exit(changeproc);
        clear;
        lines(6);
        writeln('  Current status is...',stat1);
        writeln('     On target list?..',stat2);
        yescheck;
        if yes then tgtstatus;
        if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current type is...',ttypebuf);
      yescheck;
      if yes then tgttype;
      if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current altitude is...',alt);
      yescheck;
      if yes then tgtalt;
      if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current supporting arm assigned is...',sabuf);
      yescheck;
      if yes then SAassgn;
      if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current remarks are...',rem);
      yescheck;
      if yes then remarks;
      if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current map reference is...',maprefer);
      yescheck;
      if yes then mapref;
      if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current source is...',sour);
      yescheck;
      if yes then source;
      if punchout then exit(changeproc);
      clear;
      lines(5);
      writeln('  Current air photo number is...',photonum);
      yescheck;
      if yes then afotonum;
      if punchout then exit(changeproc);
      cngproc2;
   end;
 end;




begin{change}
  mandatoryitem := false;
  changeinfo;
  if not current then retchtgt(1);
if quit then exit(change);
repeat
  punchout := false;
  clear;
  lines(2);
  writeln('      Target  ',currenttgt.tnum,' is loaded into memory');
  lines(2);
  getfile('changemenu.text');
  lines(2);
  select;
  read(ch);
  if ch in ['1','2','3','4','R','r','?'] then
  begin
    if eoln(input) then readln;
    case ch of
     '1' : displaytgt;
     '2' : changeproc;
     '3' : begin
             putinfile;
             outprocess := true;
             exit(change);
           end;
     '4','R','r' : exit(change);
     '?' : begin
             lines(2);
             getfile('changex.text');
             lines(2);
             spacebar;
           end;
    end;
   end;
  until menuloop = true;
 end;
```

```
{***********************************************************************************************}

                    (*     TARGET.TEXT      *)


   procedure buildgridmap;

   var  j : integer;

   begin
     j := 0;
     i := 0;
     while not eof(target) do
     begin
         seek(target,i);
         get(target);
         gridmap[i] := target^.tgtrec.loc;
         i := i + 1;
         j := j + 1;
         if j = 10 then
           begin
             write(dot);
             j := 0;
           end;
       end;
   end;


  procedure fetchtgt;

  var  holder, n : integer;
       first : boolean;

begin
  first := true;
  n := 0;
  range := 5;
  fetchback := false;
  finished := false;
  quit := false;
  mandatoryitem := false;
  n := 1;
  while not finished do
  begin
    lines(5);
    if grid = 2 then writeln('      ENTER GRID LOCATION')
    else writeln('      ENTER TARGET NUMBER');
    prompt;
    readln;
    if grid = 2 then checkdigit (str,ok,range)
    else checknum(str,ok);
    if quit then exit(fetchtgt);
    if fetchback then helpme := true;
    if helpme or not ok then
    begin
      finished := false;
      lines(3);
      if grid = 2 then
      begin
      getfile('tgtloc.text');
      end
      else
      begin
      getfile('tnum.text');
      end;
      lines(2);
      returnbar;
    end;
    if (flg = 2) and finished then
    begin
      clear;
      lines(5);
      if grid = 2 then
      begin
        write('    Searching for grid coordinates   ',str);
        buildgridmap;
      end
      else write('  Searching for target   ',str);
      recnum := 1;
      write(dot);
      if grid = 2 then
        begin
          for recnum := 1 to numbertgt - 1 do
          begin
            if gridmap[recnum] = str then
            begin
              lines(1);
writeln('          Target no. ',tgtmap[recnum],' has coordinates '.str);
              if first then holder := recnum;
              first := false;
              r := r + 1;
            end;
          end;
          if r = 1 then recnum := holder;
          if r > 1 then
          begin
            finished := false;
            lines(1);
         writeln('     Select the desired target number from the above list '
            lines(1);
            spacebar;
            exit(fetchtgt);
          end
     end
     else while (tgtmap[recnum] <> str) and not (recnum = numbertgt - 1) do
         recnum := recnum + 1;
      write(dot);
      if recnum = numbertgt - 1 then
      begin
        finished := false;
        error2;
      end
      else
      begin
        write(dot);
        seek(target,recnum);
        get(target);
        write(dot);
        currenttgt := target^.tgtrec;
        seek(QT,recnum);
        get(QT);
        currentQT := QT^.querrec;
        write(dot);
      end;
    end;
  end;
end;


     procedure del;

     begin
       QT^.querrec := emptyqrec;
       seek(QT,recnum);
       put(QT);
       target^.tgtrec := emptytrec;
       seek(target,recnum);
       put(target);
       tgtmap[recnum] := 'zzzzzz';
     end;


     procedure deletetgt;

     begin
       clear;
       lines(2);
       writeln('                    DELETE TARGET');
       lines(2);
       getfile('deltgt.text');
       getcntgt(1);
       if not quit then del;
     end;




     procedure caseproc;

     var  holdcar : char;
```

```
begin
  with currenttgt do
  begin
    holdcar := sa;
    case holdcar of
      '1' : saout := 'ARTY          ';
      '2' : saout := 'NGF           ';
      '3' : saout := 'AIR           ';
      '4' : saout := 'AIR, ARTY     ';
      '5' : saout := 'AIR, NGF      ';
      '6' : saout := 'ARTY, NGF     ';
      '7' : saout := 'AIR, ARTY, NGF';
      '8' : saout := 'Other         ';
      '9' : saout := 'None          ';
    end;
    holdcar := acc;
    case holdcar of
      '1' : accout := 'Confirmed';
      '2' : accout := 'Probable ';
      '3' : accout := 'Possible ';
      '4' : accout := 'Unknown  ';
    end;
    holdcar := pri;
    case holdcar of
      '1' : priout := 'I  ';
      '2' : priout := 'II ';
      '3' : priout := 'III';
      '4' : priout := 'IV ';
    end;
    holdcar := ttype;
    case holdcar of
      '1' : ttypeout := 'TANK';
      '2' : ttypeout := 'SQAD';
      '3' : ttypeout := 'INST';
      '4' : ttypeout := 'CBAT';
      '5' : ttypeout := 'OP  ';
      '6' : ttypeout := 'TERR';
      '7' : ttypeout := 'VEH ';
      '8' : ttypeout := 'FORT';
      '9' : ttypeout := 'MISC';
    end;
    holdcar := stat;
    case holdcar of
      '1','2' : begin
                  stat1 := 'ACTIVE  ';
                  stat2 := 'YES';
                end;
      '3','4' : begin
                  stat1 := 'ACTIVE  ';
                  stat2 := 'NO ';
                end;
      '5','6' : begin
                  stat1 := 'INACTIVE';
```

```
                                stat2 := 'NO ';
                              end;
                  end;
                end;
              end;



          procedure maincard;


          begin
            clear;
            with currenttgt do
              begin
writeln(dots);
writeln('                        : TARGET NO. ',tnum,':');
writeln('                        ------------------------');
writeln('Location: ',loc,'        alt: ',alt,'          Type: ',ttypecur);
writeln('Description: ',desc);
writeln('Class     : ',class,'                        Status: ',stat1);
writeln('Priority : ',pricur,'                      Tgt List?: ',stat2);
writeln('SA Assng : ',sacur);
writeln('Source of Tgt: ',sour);
writeln('DTG Activated: ',DTGact);
writeln('Photo No: ',photonum,'          Map Ref : ',mapref);
writeln('Photo Coord: ',photocord,'     Accuracy: ',accur);
writeln('Remarks: ',rem);
              end;
          end;




procedure BDAcard;

var holdcar : char;
    damrepcur : string[11];
    damasscur : string[11];

          begin
            damasscur := '           ';
            damrepcur := '           ';
writeln('--------------------SURVEILLANCE--------------------------');
writeln('              Firing      No/type          Damage        Damage');
writeln('     DTG       Unit        Rounds         Reported      Assessed');
writeln('     ---      ------      -------         --------      --------');
            for i := 1 to 3 do
              if (currenttgt.flag[i + 16] = off) then with currenttgt.BDA[i] d
                begin
                  holdcar := damrep;
                  case holdcar of
                  '1' : damrepcur := 'DAMAGED    ';
                  '2' : damrepcur := 'DESTROYED  ';
          '3' : damrepout := 'INTERDICTED';
          '4' : damrepout := 'HARASSED   ';
          '5' : damrepout := 'NEUTRALIZED';
          '6' : damrepout := 'ILLUMINATED';
          '7' : damrepout := 'Unobserved ';
          '8' : damrepout := 'Unknown    ';
          '9' : damrepout := '           ';
          end;
          holdchar := damass;
          case holdchar of
          '1' : damassout := 'DAMAGED    ',
          '2' : damassout := 'DESTROYED  ';
          '3' : damassout := 'INTERDICTED';
          '4' : damassout := 'HARASSED   ';
          '5' : damassout := 'NEUTRALIZED';
          '6' : damassout := 'ILLUMINATED';
          '7' : damassout := 'Unobserved ';
          '8' : damassout := 'Unknown    ';
          '9' : damassout := '           ';
          end;
write(' ',DTGsurv:7,'   ',fireunit:5,'    ',atrocs:1[illegible];
writeln('       ',damrepout:11,'   ',damassout);
writeln('BDA: ',BDAtext);
      end;
   end;


   procedure displaytgt;

     begin
       with currenttgt do
         begin
           caseproc;
           maincard;
           if stat in ['1','3','5'] then BDAcard;
         end;
        write('...........',returner,'      ...........');
        repeat
          read(ch);
        until eoln(input);
      end;



   procedure display;

   begin
     currenttgt := emptyTrec;
     finished := false;
     quit := false;
     repeat;
       clear;
       lines(2);
       writeln('                    DISPLAY TARGET CARD');
```

```
    lines(1);
    writeln('                The options are:');
    lines(1);
    writeln('                        1.  Target number');
    writeln('                        2.  Grid location');
    writeln('                        3.  Information');
    writeln('                        4.',return);
    lines(1);
    select;
    read(ch);
    if ch in ['1','2','3','4','R','r','Q','q','?'] then
    begin
      if eoln(input) then readln;
      case ch of
        '1' : fetchtgt(1);
        '2' : fetchtgt(2);
        '3','?' : begin
                    lines(1);
                    getfile('disptgt.text');
                    lines(1);
                    spacebar;
                  end;
        '4','R','r','Q','q' : exit(display);
      end
    end
    else menuerror;
  until finished or quit;
  if not quit then displaytgt;
end;




procedure putinfile;


begin
  seek(target,recnum);
  get(target);
  target^.tgtrec := currenttgt;
  seek(target,recnum);
  put(target);
  tgtmap[recnum] := target^.tgtrec.tnum;
  seek(QT,recnum);
  get(QT);
  QT^.querrec := currentQT;
  seek(QT,recnum);
  put(QT);
  currenttgt := emptyTrec;
  currentQT := emptyQrec;
end;
```

```
procedure process;

begin
if outprocess then exit(process);
repeat
  clear;
  writeln(dots);
  lines(3);
  writeln('      Target Input Has Been Completed');
  lines(2);
  writeln('       Your options are:');
  lines(1);
  writeln('              1.  Write target to the file');
  writeln('              2.  Display information on the screen');
  writeln('              3.  Change part of the information');
  writeln('              4.  Delete the target');
  if (currentgt.stat = '1') or (currentgt.stat = '3') then
  writeln('              5.  Add battle damage assessment (BDA)');
  lines(1);
  select;
  read (ch);
  if ch in ['1'..'5'] then
   begin
    if eoln(input) then readln;
    finished := true;
    case ch of
     '1' : putinfile;
     '2' : begin
             finished := false;
             displaytgt;
           end;
     '3' : begin
             current := true;
             change;
             if outprocess then exit(process);
           end;
     '4' : del;
     '5' : if (currentgt.stat = '1') or (currentgt.stat = '3') then
             begin
               BDAcounter := 1;
               current := true;
               newBDA;
               finished := false;
             end
            else menuerror;
    end;
   end
  else if ch = '?' then
      begin
         finished := false;
```

```
          clear;
          lines(8);
          getfile('procinfo.text');
          lines(1);
          spacebar;
        end
     else tenderror;
  until finished = true;
end;


     procedure checkdone;

     var  count,m : integer;

     begin
       count := 0;
       for m := 1 to numtocheck do
         if currenttgt.flag[m] <> off then count := count + 1;
       if count = 7 then
       begin
         clear;
         lines(2);
         writeln('    Entry of target number ', currenttgt.tnum,' is complet
         lines(2);
         spacebar;
         done := true;
       end
       else begin
              clear;
              lines(8);
writeln('A number of items have not been completed for target ',currenttgt
              repeat
              lines (2);
              writeln('        The options are:');
              lines(1);
              writeln('            1.  Continue working on target');
              writeln('            2.  Stop working on target');
              writeln('            3.  Make changes to the target data');
              lines(1);
              select;
              read(ch);
              if ch in ['1','2','3','?'] then
                begin
                  if eoln(input) then readln;
                  case ch of
                    '1' : exit(checkdone);
                    '2' : begin
                            done := true;
                            exit(checkdone);
                            end;
                    '3' : begin
                            done := true;
```

```
                                        current := true;
                                        change;
                                        exit(checkone);
                                        end;
                          '?' : begin
                                    clear;
                                    lines(2);
                                     getfile('ckionex.text');
                                     lines(2);
                                     returnbar;
                               end
                           end
                      end
                      else menuerror;
                      clear;
                  until menuloop = true;
               end;
           end;


       procedure addtgtinfo;

egin
 clear;
 writeln(dots);
 lines(2);
 getfile('addtgtinfo.text');
 lines(2);
 spacebar;
 clear;
 lines(2);
 getfile('addtinf1.text');
 lines(2);
nd;



         procedure newtarget;

         var j, t : integer;

         begin
            recnum := 0;
            while tgtmap[recnum] <> '000000' do
            begin
               recnum := recnum + 1;
               if recnum = numbertgt - 1 then
               begin
                  clear;
                  lines(11);
                  writeln('          FILE FULL');
                  lines(2);
                  writeln('          Targets must be deleted in order to continue');
            lines(1);
            spacebar;
            done := true;
            exit(newtarget);
          end;
        end;
        seek(QT,recnum);
        get(QT);
        currentQT := QT^.querrec;
        seek(target,recnum);
        get(target);
        currenttgt := target^.tgtrec;
        for t := 1 to numobservar do
          currenttgt.flag[t] := on;
      end;




      procedure edittarget;

      var  i : integer;

begin
  numbcheck := 15;
  mandatoryitem := true;
  current := false;
  outprocess := false;
  finished := false;
  done := false;
  quit := false;
  ok := true;
  clear;
  if first then
    begin
      writeln(dots);
      lines(1);
      getfile('pretgt.text');
      prompt;
      first := false;
    end
  else
    begin
      lines(5);
writeln('  **  For information, type a ?....To continue, press RETURN  **
      prompt;
    end;
  repeat
  read(ch);
  if ch = '?' then
       begin
        edittginfo;
        returnbar;
      end;
 until eoln(input);
newtarget;
if done then exit(addtarget);
while not done do
begin
  for n := 1 to numbervar - 0 do
   begin
     if quit then exit(addtarget);
     if currenttgt.flag[n] = on then
     begin
       case n of
       1 : tgtnum;
       2 : tgtloc;
       3 : tgtdesc;
       4 : tgtclass;
       5 : tgtpri;
       6 : tgtstatus;
       7 : tgttype;
       8 : begin
             mandatoryitem := false;
             tgtalt;
           end;
       9 : unassgn;
      10 : remarks;
      11 : mapref;
      12 : source;
      13 : photonum;
      14 : photogrid;
      15 : DTGactive;
      16 : tgtaccuracy;
     end;{case}
   end;
   finished := false;
   end; {for}
   if flg <> 0 then checkdone
   else done := true;
  end; {while}
  process;
end;



      procedure tgtmenu;

      begin
        clear;
        writeln(stars);
        lines(2);
        writeln('            Working On The Target File');
        lines(1);
        writeln('      The options are:');
        lines(1);
        writeln('              1.  Add a target');
        writeln('               2.  Change a target');
        writeln('               3.  Display a target');
        writeln('               4.  Enter a target surveillance');
        writeln('               5.  Delete a target');
        writeln('               6.  List targets currently in file');
        writeln('               7.',return);
        lines(2);
      end;


procedure listcheck;

var   j : integer;

    begin
      j := 4;
      clear;
      lines(2);
      writeln('                 LIST OF CURRENT TARGETS IN FILE');
      lines(2);
      for i := 1 to numbertgt do
        if tgtmap[i] <> 'bbbbbb' then
          begin
            write('        ',tgtmap[i]);
            j := j - 1;
            if j = 0 then
              begin
                writeln;
                j := 4;
              end;
          end;
      lines(2);
      writeln('                   END OF TARGET LIST');
      lines(1);
      spacebar;
    end;




begin {targetmod}
  nostring := '      ';
  nochar := ' ';
  first := true;
  mandatoryitem := false;
  menuloop := false;
  menuchar := ['1','2','3','4','5','6','7','?','h','r','a','b','c','a','d'
      repeat
       curr nt := false;
       tgtmenu;
       select;
       read(ch);
       if ch in menuchar then
       begin
      if eoln (input) then readln;
      case ch of
      '1','A','a' : addtarget;
      '2','C','c' : change;
      '3','D','d' : display;
      '4' : begin
              menuforyitem := false;
              newsdn;
            end;
      '5' : deletetgt;
      '6' : listcheck;
      '7','X','x' : exit(targetmod);
      '?' : begin
              clear;
              lines(6);
              getfile('tgthook.text');
              lines(6);
              spacebar;
            end
      end
    end
    else menuerror;
  until menutop = true;
end;
```

```
                (*    INITIALIZE    *)


segment procedure initialize;

    var
          menchr : mvalue;
          goback : boolean;

     procedure initinfo;

     begin
clear;
writeln(stars);
lines(1);
writeln('   If the system has not been initialized when the program is');
writeln('started, it will initialize automatically and create the target'
writeln('files on the diskette. This procedure allows you to re-initializ
writeln('the system for a new operation or to restart target information'
writeln('operations from a fresh start.');
lines(2);
writeln('   Option 2 will delete all the current files and allow you to'
writeln('start out from the beginning. The files you delete are not');
writeln('recoverable. Be sure you want to delete all of your files befor
writeln('you select option 2. Use option 3 to return to the main');
writeln('command menu. Pressing the return key will return you to the');
writeln('initialize option menu.');
lines(2);
returnbar;
end;


procedure initfile;

     var  i, j : integer;

begin {initfile}
  restart := true;
  write(dot);
  j := 1;
  write(dot);
  close(target,purge);
  rewrite(target,'#5:targetfile.data');
  for i := 1 to numbertgt do
    begin
      j := j + 1;
      if j = 15 then
      begin
        write(dot);
        j := 1;
      end;
      target^.tgtrec := emptyTrec;
      put(target);
    end;
  close(target,lock);
  reset(target,'#5:targetfile.data');
  write(dot);
  close(QT,purge);
  rewrite(QT,'#5:queryfile.data');
  write(dot);
  for i := 1 to numbertgt do
    begin
      j := j + 1;
      if j = 15 then
      begin
        write(dot);
        j := 1;
      end;
      QT^.querrec := emptyQrec;
      put(QT);
    end;
  close(QT,lock);
  reset(QT,'#5:queryfile.data');
  write(dot);
end;


  procedure reinitialize;

  begin {reinit}
    goback := false;
    clear;
    lines(4);
writeln('       THIS PROCEDURE WILL DELETE ALL TARGET FILES.');
writeln(dots);
lines(1);
writeln(chr(7));
writeln('        The options are:');
lines(1);
writeln('               1.  Return to Main Command Menu');
lines(1);
writeln('               2.  Reinitialize System');
lines(1);
writeln('               3.  Information');
lines(1);
    select;
    repeat
    read(ch);
    if eoln (input) then
    begin
      goback := true;
      exit(reinitialize);
    end;
    if ch in ['1'..'3'] then
    begin
      case ch of
      '1' :  exit(reinitialize);
      '2' : begin
```

```
            lines(2);
            write('        System initializing....');
            write(dot);
            initfile;
            lines(2);
            exit(reinitialize);
          end;
      '3','?' : initinfo;
       end
    end
    else menuerror;
    until goback = true;
end;{reinit}


          begin {initialize}
            goback := false;
            menchr := ['1','2','3','R','r','?'];
            if filecheck then
              begin
                initfile;
                exit(initialize);
              end;
          repeat
            clear;
            writeln(dots);
            lines(2);
            writeln('               Initializing the System');
            lines(2);
            writeln('      The options are:');
            lines(1);
            writeln('            1.  Information');
            lines(1);
            writeln('            2.  Initialize the System');
            lines(1);
            writeln('            3.'.return');
            lines(2);
            select;
            read(ch);
            if ch in menchar then
            begin
              if eoln (input) then readln;
              case ch of
              '1','?' : initinfo;
              '2' : begin
                      reinitialize;
                      if not goback then exit(initialize);
                    end;
              '3','R','r' : exit(initialize);
              end
            end
          else menuerror;
          until menuloop = true;
        end;
```

```
segment procedure inform;


var
menchar : tvalue;



   procedure informenu;

   begin
   clear;
   writeln(dots);
   lines(2);
   writeln('  This section provides information about the following:');
   lines(2);
   writeln('              1.  How to Operate the System');
   writeln('              2.  Security Requirements');
   writeln('              3.  Target Classifications');
   writeln('              4.  Target priorities');
   writeln('              5.  Target Analysis Guidelines');
   writeln('              6.',return);
   lines(2);
   end;



   procedure userinst;

   begin
     clear;
     writeln('           System Operation Instructions');
     writeln(dots);
     lines(12);
     writeln('  TO BE INSERTED');
     lines(5);
     spacebar;
   end;



   procedure format1; forward;
   procedure formatoptions; forward;

   procedure formats;

   begin
     clear;
     lines(1);
     writeln('            Formats Used in the Displays');
     writeln(dots);
     lines(1);
riteln('    There are two basic formats used in the target information ');
```

```
writeln('system. The first is a replica of the current target card as');
writeln('specified in FMFM 7-1 (Fire Support Coordination). All of the ');
writeln('information about a particular target, including target');
writeln('surveillance, can be displayed on the screen or printed on the');
writeln('line printer.');
lines(1);
writeln('     The second type of display is the target listing. This');
writeln('listing contains the most important items from the target');
writeln('data and is used primarily by the supporting arms members of');
writeln('the FSCC. This listing is available in many different forms,');
writeln('from the Target List to a special listing of particular target');
writeln('characteristics.');
        lines(2);
        spacebar;
        format1;
      end;


      procedure format1;

      begin
      clear;
      lines(1);
      writeln('     One procedure in this system allows you to display or')
      writeln('print a list of targets with a combination of the following
      writeln('parameters: status, priority, classification, supporting');
      writeln('arm assigned, target type and accuracy. For example, you');
      writeln('could obtain a list of all fortifications targets assigned
      writeln('to naval gunfire for the naval gunfire officer, all priorit
      writeln('II and III counterfire targets assigned to artillery for th
      writeln('artillery officer or all active SEAD targets for the air');
      writeln('officer.');
      lines(2);
      writeln('     A third item that can be displayed or printed is the');
      writeln('target bulletin (TARBUL). The system automatically keeps');
      writeln('track of all transactions for the Target List and allows');
      writeln('you to print a formatted TARBUL suitable for transmission.
      writeln('It uses the standard format of targets added to the list,')
      writeln('deleted from the list,targets changed and important target
      writeln('surveillances.');
      lines(1);
      spacebar;
      formatoptions;
      end;


      procedure Tcarddis; forward;
      procedure Tlistdis; forward;
      procedure Tbuldis; forward;
      procedure formopmenu; forward;



      procedure formatoptions;
```

```
    begin
    repeat
      formopmenu;
      select;
      read(ch);
      if ch in menchar then
      begin
        if eoln(input) then readln;
        case ch of
           '1' : Tcardlis;
           '2' : Tlistlis;
           '3' : Tbullis;
           '4','R','r' : exit(formatoptions);
           '5','6' : menuerror;
           '?' : begin
                 lines(2);
riteln('  The three options above will display the computer generated');
riteln('formats for the target card, the target list and the target');
riteln('bulletin. Choose one of these three options or type option 4');
riteln('to continue operations and return to the previous menu.');
                 lines(2);
                 returnbar;
                 end
           end
        end
        else menuerror;
    until menuloop = true;
   end;



    procedure formopmenu;

    begin
      clear;
      writeln(dots);
      lines(2);
      writeln('            The Display Options are:');
      lines(3);
      writeln('                 1.  Target Card');
      writeln('                 2.  Target Listing');
      writeln('                 3.  Target Bulletin (TARBUL)');
      writeln('                 4.',return);
      lines(2)
      end;


      procedure Tcardadd;

      begin
writeln('Photo No:                    Map Ref   : IRAN 4577-IV');
writeln('Photo Coord:                 Accuracy  : CONFIRMED');
writeln('Remarks: first tank sighting in sector IV. Attack w/ rockeye');
lines(1);
writeln('-------------------SURVEILLANCE-------------------------');
writeln('            Firing      NO/type         Damage      Damage');
writeln('   DTG       Unit        Rounds         Reported    Assessed');
writeln('   ---      ------      -------        --------    --------');
writeln(' 121630Z    2 F/A-18    2  D-22        DESTROYED   DESTROYED',;
writeln('BDA: Both tanks confirmed by AO. No AAA coverage on tgt.');
       lines(1);
       end;


     procedure Tcard11s;

     begin
       clear;
writeln(dots);
writeln('                 : TARGET NO.  ADØØ12 :');
writeln('                 ----------------------');
writeln('Location: 34567575         Alt: 9Ø    Type: TANK');
writeln('Description: 2 T-62 TANKS IN OPEN FIELD');
writeln('Class    : A                        Status : ACTIVE');
writeln('Priority : I                      Tgt List?: YES');
writeln('SAassgn  : AIR');
lines(1);
writeln('Source of Tgt: AIR OBSERVER OV-1Ø');
writeln('DTG Activated: 1216ØØZ');
       Tcarda11;
       spacebar;
       end;



     procedure T11st11s;

     begin
       clear;
writeln(stars);
lines(2);
writeln('              TARGET LISTING');
lines(1);
writeln('TGT NO   CL  PRI LOCATION  ALT    SAASG    DESCRIPTION');
writeln('------   --  --- --------  ----  ------    -----------');
writeln('ADØØ21*  A    II 34566577   9Ø    AIR     2 T-62 TANKS IN OPEN');
writeln('ADØØ24*  A     I 34524355  1ØØ    NGF     FORTIFIED BUNKER COMPL
writeln('ADØØ33*  B    II 34556654   45   ARTY     PLT OF T-1ØØ AT GUNS');
writeln('ADØØ54   D    IV 34555566   1Ø    NGF     BEACH FORTIFICATIONS')
writeln('ADØØ55*  E    IV 34455776   5Ø   NONE     SCHOOL BUILDINGS');
writeln('NAØ211   C   III 43226555    Ø    NGF     BEACH FORTIFICATIONS',
writeln('ADØØ37*  A     I 34778545  12Ø    AIR     PLT ZSU-23-4 IN TREES'
writeln('ADØØ57*  E    IV 33507655  145   NONE     RAIL/SUPPLY DEPOT');
writeln('ADØØ66   B   III 3444690Ø   2Ø   ARTY     PLT DUG IN TRENCHLINE'
writeln('ADØØ68*  A     I 3422659Ø   7Ø    AIR     12 VEHICLES ALONG ROAD
       lines(2);
```

```
      writeln('   NOTE: * indicates target list');
      lines(1);
      spacebar;
    end;


    procedure Tbuildisl;

    begin
lear;
ines(2);
riteln('   3.  CANCELLED TARGETS');
riteln('       AD0034, AD0035, AD0056, AD00122, NA0201');
riteln('       AD0043, AD0097, AD0100');
ines(1);
riteln('   4.  REACTIVATED TARGETS');
riteln('       AD0077, AD0103');
ines(1);
riteln('   5.  CLASSIFICATION/PRIORITY CHANGE');
riteln('        AD0053  A  II');
riteln('        AD0054  C  IV');
riteln('        AD0079  D  III');
riteln('        AD0107  E   IV');
riteln('        AD0121  E    I');
riteln('        AD0221  A   II');
ines(1);
riteln('                                                   CLASSIFICATION ');
ines(2);
pacebar;
    end;


    procedure Tbuildis;

    begin
    clear;
writeln('                    TARGET BULLETIN');
lines(1);
writeln('CLASSIFICATION');
lines(1);
writeln('   DTG  :  121630Z');
writeln('   FROM :  CFL  (CTF32.1.1)');
writeln('   TO   :  DISTRIBUTION');
lines(1);
writeln('   SUBJ :  TARBUL NUMBER 12');
lines(1);
writeln('   1.  NEW TARGETS');
writeln('       AD0134  34555544  FORTIFICATIONS     A  I');
writeln('       AD0135  34526577  2 T-34 TANKS       A  I');
writeln('       AD0136  34567060  BUNKER COMPLEX     E  III');
lines(1);
writeln('   2.  BDA');
writeln('       AD0078  80% damaged by air strike');
writeln('       AD0083  Destroyed');
writeln('         ADW115  Partially damaged by artillery');
lines(1);
spacebar;
Tbuildisl;
end;



      procedure tgtinfo;

      begin
        clear;
        writeln('           Target listing Information');
        writeln(dots);
        lines(1);
        getfile('queryinfo.text');
        lines(1);
        spacebar;
      end;

       procedure version1; forward;

       procedure versadd;

       begin
writeln('      CRT CONSOLE: Datamedia Elite 2500');
writeln('         LANGUAGE: Pascal');
writeln(' IMPLEMENTATION: UCSD Pascal (version 1.4b)');
writeln('           DESIGN: LtCol R. J. Coulter, USMC');
writeln('      PROGRAMMING: LtCol R. J. Coulter, USMC');
lines(1);
spacebar;
       end;

       procedure version;

       begin
       clear;
       writeln(dots);
writeln('  Microcomputer System for Target Information  (MISTI)');
writeln('  ------------------------------------------------------');
writeln('                    Version 1.0');
lines(1);
writeln(' A prototype microcomputer data base operation system for the')
writeln('      target information section of the Marine Corps fire suppor
writeln('      coordination center. It is the result of a masters thesis'
writeln('      submitted at the Naval Postgraduate School.');
lines(1);
writeln('         LOCATION: Department of Computer Science');
writeln('                   Naval Postgraduate School');
writeln('                   Monterey, California');
writeln('             DATE: 15 June 1981');
writeln('SOURCE COMPUTER: Altos ACS 8000-1');
writeln('OBJECT COMPUTER: Altos ACS 8000-1');
```

```
ersall;
ersionl;
nd;


rocedure versionl;

egin
lear;
ines(4);
riteln(' System supports upper and lower case. Character delete key ');
riteln('     is <rubout> key. Input terminator is <return> key.');
ines(1);
riteln(' FOR INFORMATION:');
ines(1);
riteln(' Professor Lyle A. Cox,Jr.');
riteln('           Naval Postgraduate School');
riteln('           Monterey, California 93940');
riteln('           408-646-2449');
ines(1);
riteln('     LtCol Donald J. Coulter, USMC');
riteln('           Development Center');
riteln('           MCDEC');
riteln('           Quantico, Virginia 22134');
riteln('           PHONE');
lines(5);
spacebar;
end;

   procedure sysopmenu;

   begin
     clear;
     writeln(dots);
     lines(1);
     writeln('               System Operation');
     lines(1);
     writeln('      The options are:');
     lines(1);
     writeln('         1.  Instructions for the User');
     writeln('         2.  Formats Used in Displays');
     writeln('         3.  Obtaining Information about Targets');
     writeln('         4.  System Technical Information';
     writeln('         5.',return);
     lines(2);
   end;



   procedure systemop;

   begin
     repeat
       sysopmenu;
```

```
        select;
        read(ch);
        if ch in menchar then
        begin
          if eoln (input) then readln;
          case ch of
            '1' : userinst;
            '2' : formats;
            '3' : tgtinfo;
            '4' : version;
            '5','R','r' : exit(systemop);
            '6' : menuerror;
            '?' : begin
                    lines(1);
writeln('    Five options concerning system operation are provided in');
writeln('in the above menu. Select the item you want from these options');
writeln('and type that number on the keyboard. If you do not desire any');
writeln('information on system operations, then use option 5 to return');
writeln('to the previous menu');
                    lines(1);
                    returnbar;
                    end
            end
          end
          else menuerror;
        until menuloop = true;
      end;




     procedure security;

     begin
     clear;
     writeln('           Security Guidelines');
     writeln(dots);
     lines(10);
     writeln('   TO BE IMPLEMENTED');
     lines(7);
     spacebar;
     end;


     procedure tgtclass;

     begin
       clear;
       lines(1);
       getfile('class.text');
       lines(1);
       spacebar;
     end;
```

```
    procedure tgtpri;

    begin
      clear;
      lines(1);
      getfile('priority.text');
      lines(3);
      spacebar;
    end;


    procedure anal1; forward;
    procedure anal2; forward;
    procedure anal3; forward;
    procedure anal4; forward;
    procedure anal5; forward;

    procedure tgtanal;

    begin
    clear;
    lines(1);
    writeln('                Target Analysis Guidelines');
    writeln(dots);
    lines(1);
riteln('  The following format ensures a logical and orderly examination');
riteln('of all factors to determine the best method of attack of a target.');
ines(1);
riteln('Situation of opposing forces:');
riteln('------------------------------');
riteln('Enemy situation...include information that will aid target analysis.'
ines(1);
riteln('Friendly situation...information that will aid attack of the target.'
ines(1);
riteln('Target characteristics:');
riteln('-------------------------');
riteln('Target description....type(personnel, materiel, terrain), number');
riteln('                  of personnel, quantity of materiel and activity.');
ines(1);
riteln('Vulnerability...type and amount of cover, type of materiel, type');
riteln('             of construction, mobility and density of personnel');
riteln('             and material.');
ines(1);
pacebar;
nal1;
nd;

rocedure anal1;

egin
lear;
ines(1);
riteln('Physical location....grid reference, altitude of target, location');
```

```
writeln('              of friendly forces and terrain features.');
lines(1);
writeln('Accuracy....of the target location and the agency reporting the');
writeln('          target.');
lines(1);
writeln('Size of area...dimensions and shape of the target area and the');
writeln('          distribution of personnel and materiel in the area.'
lines(1);
writeln('Terrain and weather...brief analysis of terrain and weather');
writeln('           in the target area. Include any terrain features');
writeln('           which affect the means and method of attack.');
lines(2);
writeln('Target Capabilities:');
writeln('--------------------');
writeln('    The capabilities of the target as they affect the accomplishm
writeln('  of the mission of the supported unit. Show how a terrain featur
writeln('  affects enemy capabilities.');
lines(1);
spacebar;
anal2;
end;

procedure anal2;

begin
clear;
lines(2);
writeln('Other Factors:');
writeln('--------------');
writeln('  How do the following affect the choice of firepower, method'
writeln('of attack and delivery means?');
lines(1);
writeln('Urgency of attack....determined by the type of target (static or
writeln('          fleeting) and its capabilities.');
lines(1);
writeln('Enemy countermeasures...ability of the enemy to minimize the');
writeln('          effects of firepower, prevent delivery of supporting'
writeln('          arms and bring countermeasures against delivery means
writeln('          after attack.');
lines(1);
writeln('Enemy discipline...factors which will aid in determining the');
writeln('        amount of firepower required to neutralize the morale');
writeln('        and discipline of enemy personnel.');
lines(1);
writeln('Obstacles....considerations concerning the desirability of ');
writeln('       creating obstacles by attacking the target.'):
lines(1);
spacebar;
anal3;
end;


procedure anal3;
```

```
egin
lear;
ines(2);
riteln('Civilian casualties...approximate number of civilians in the');
riteln('      target area and the estimated effect of causing excessive');
riteln('      casualties.');
ines(1);
riteln('Suprise....methods desired to obtain surprise, including least');
riteln('           expected time of attack, means of delivery and');
riteln('           restrictions on artillery preregistration.');
ines(1);
riteln('Means of Attack:');
riteln('----------------');
riteln('      All available types of firepower and maneuver elements');
riteln('with which it is practical to attack the target as well as the');
riteln('most practicable delivery means in each case.');
ines(1);
riteln('Analysis of Means of Attack:');
riteln('----------------------------');
riteln('  The effect of each means of attack on the target characteristics,');
riteln('target capabilities and other factors. For each means of attack');
riteln('include the factors on the next frame.');
ines(1);
pacebar;
nal4;
nd;

rocedure anal4;

egin
lear;
ines(2);
riteln('1.  Location of center of impact which will give best effect.');
ines(1);
riteln('2.  Effect of available supply rate.');
ines(1);
riteln('3.  Estimate of enemy casualties and material damage.');
ines(1);
riteln('4.  Estimate of civilian casulties.');
ines(1);
riteln('5.  Estimate of obstacles created.');
ines(1);
riteln('6.  Precautions required for friendly troops.');
ines(1);
ines(1);
riteln('Comparison of Means of Attack:');
riteln('------------------------------');
riteln('   The outstanding advantages and disadvantages of each means');
riteln('of attack and determine which offers the greatest promise of');
riteln('success.');
ines(1);
pacebar;
nal5;
nd;
```

```
procedure analp;

begin
clear;
lines(2);
writeln('Decision or Recommendation:');
writeln('----------------------------');
lines(1);
writeln('1.  Type and amount of firepower and delivery means.');
lines(1);
writeln('2.  Units to fire.');
lines(1);
writeln('3.  Grid location and altitude of desired center of impact.');
lines(1);
writeln('4.  Time of attack.');
lines(1);
writeln('5.  Safety precautions, special coordination and warnings.');
lines(1);
writeln('6.  Method of determining post-strike analysis.');
lines(4);
returnbar;
end;
```

```
      begin {inform}
         menchar := ['1','2','3','4','5','6','M','m','?'];
       repeat
         informmenu1;
         select;
         read(ch);
         if ch in menchar then
            begin
            if eoln (input) then readln;
            case ch of
                '1' : systemop;
                '2' : security;
                '3' : tgtclass;
                '4' : tgtpri;
                '5' : tgtanal;
                'M','m','6' : exit (inform);
                '?' : begin
   lines(1);
   writeln('  Six options are provided in the above menu. Select');
   writeln('the item you want from these options. Type that number');
   writeln('on the keyboard and the system will respond with the');
   writeln('desired information. If you do not desire any information'
   writeln('then type option 6 to return to the main command menu.');
   lines(1);
   returnbar;
```

```
                end
              end
              end
            else menuerror;
            until finished = true;
   end;
```

```
(*              UTILITY.PAKT          *)

segment procedure utility;

    var
          menunum : mvalue;
          t : integer;

 procedure changePW;


procedure PWinst1;

  begin
  lines(1);
writeln('     The password is changed by deleting one password, substitut
writeln('an other and writing the new password to the diskette file. At
writeln('end of an operation it is suggested that all passwords be delet
writeln('This will return them to their original value MARINE. Initiali
writeln('the system will accomplish this.');
   lines(1);
   end;


 procedure PWinst;

     begin
       clear;
       lines(3);
writeln('      There are 5 passwords in the target information system.');
writeln('One is the system password which cannot be changed. This will')
writeln('ensure that at least one of the passwords will always work. It')
writeln(' is the name of the system designer, COULTER. The other four');
writeln('passwords can be input by the TIO to allow the TIS personnel');
writeln('to have exclusive access to the target file.');
lines(1);
writeln('     Initially, these 4 user passwords are MARINE and remain the
writeln('until changed by this procedure. The password can consist of u
writeln('to 12 letters or numbers. Examples of passwords could be the T
writeln('personnel last names (JONES, PARKER, Smith), social security '
writeln('numbers (249774545, 211453446) or an, alphabetical-numerical c
       PWinst1;
       spacebar;
    end;


procedure currentPW;

begin
  clear;
  lines(4);
  writeln('          CURRENT PASSWORDS');
  writeln('           -----------------');
  lines(1);
  writeln('           1.   ',userid[1]);
  writeln('           2.   ',userid[2]);
  writeln('           3.   ',userid[3]);
  writeln('           4.   ',userid[4]);
end;



   procedure PWchange;

   var  item : integer;
        go : boolean;
        newPW : string[12];

   begin
     go := false;
     currentPW;
     writeln('          5.',return);
     lines(2);
     writeln('       ENTER THE NUMBER OF THE PASSWORD TO BE CHANGED');
     prompt;
     read(item);
     if item in [1..5] then
     begin
       if eoln(input) then readln;
       if item = 5 then exit(PWchange);
       repeat
         lines(1);
         writeln('       ENTER NEW PASSWORD......do not exceed 12 characters'
         prompt;
         readln(newPW);
         if length(newPW) > 12 then
         begin

         {    function   }

           lines(1);
 writeln('     The password can be up to 12 letters or numbers in length');
writeln('such as a name, SSN or letter-number combination. Enter the');
writeln('characters in the space after the system prompt and press the');
writeln('RETURN key. The prior password will be automatically erased and th
writeln('new password substituted for it. If you just press the RETURN key,
writeln('the prior password will remain.');
          lines(1);
        end
        else go := true;
        until go = true;
        end
        else if length(newPW) = 0 then exit(PWchange);
        lines(2);
        writeln('       Password Changed');
        lines(3);
   returnbar;
 end;





begin {changePW}
  repeat
    clear;
    lines(2);
    writeln('                  CHANGING THE PASSWORD');
    writeln(dots);
    lines(1);
    writeln('          The options are:');
    lines(1);
    writeln('            1.  Instructions for changing passwords');
    writeln('            2.  Display current passwords');
    writeln('            3.  Change password');
    writeln('            4.',return);
    lines(1);
    select;
    read(ch);
    if ch in menunum then
    begin
      if eoln(input) then readln;
      case ch of
        '1','?' : PWinst;
        '2' : begin
                currentPW;
                lines(2);
                spacebar;
              end;
        '3' : PWchange;
        '4','R','r' : exit(changePW);
        '5', '6', '7', '8' : menuerror;
      end;
    end
    else menuerror;
  until menuloop = true;
end;


procedure tarcul;

      var  tbnum : integer;


procedure dispTAPFUL;

  begin
    clear;
    lines(2);
    write('      Processing TARBUL information....');
    for i := 1 to 12 do
    begin
      write(dot);
      delay;
    end;
    clear;
    writeln('                       DUMMY TARBUL');
    lines(1);
    writeln('CLASSIFICATION');
    lines(1);
    writeln('    DTG  :');
    writeln('    FROM :');
    writeln('    TO   : DISTRIBUTION');
    writeln('    SUBJ :TARBUL NUMBER',tbnum);
    lines(1);
    writeln('    1. NEW TARGETS',
    writeln('          (Tgt no., location, description, class, priority)');
    writeln('    2. BDA');
    writeln('          (Tgt no., BDA)');
    writeln('    3. CANCELLED TARGET');
    writeln('          (Tgt no.)');
    writeln('    4. REACTIVATED TARGET');
    writeln('          (Tgt no.)');
    writeln('    5. CLASSIFICATION/PRIORITY CHANGE');
    writeln('          (Tgtno., class, priority)',
    lines(1);
    writeln('                                                   CLASSIFICATION');
    lines(1);
    spacebar;
  end;


procedure printTARBUL;

var  i, amount : integer;

begin
  amount := 0;
  clear;
  lines(2);
  writeln('  Processing and printing the TARBUL will take');
  writeln('             approximately x minutes.');
  lines(2);
  writeln('  Please enter the number of copies desired and');
  writeln('               press the RETURN key.');
  lines(1);
  prompt;
  readln(amount);
  if amount = 0 then exit(printTARBUL);
  lines(3);
  {   process   }
  writeln('    Processing file...');
  lines(2);
  for i := 1 to amount do
    begin
      delay;
      writeln('    Printing copy ',i);
      {print copy}
    end;
  lines(4);
  writeln('                  **  Printing Complete  **');
  lines(1);
  writeln('          Ensure that a new TARBUL file is created.');
  writeln('    **  Select option 4 on the following menu  **');
  lines(2);
  spacebar;
end;




procedure numTARBUL;


begin
  tbnum := 0;
  clear;
  lines(2);
  writeln('             TARBUL NUMBER');
  lines(1);
writeln('      This procedure restarts the tarbul file and assigns the');
writeln('next number in the TARBUL sequence to the file. This procedure');
writeln('is used only once...as soon as the control of the target list');
writeln('is passed ashore by the SACC final TARBUL.');
lines(2);
writeln('        ENTER TARBUL NUMBER AND PRESS RETURN');
  prompt;
  readln(tbnum);
  if tbnum = 0 then exit(numTARBUL);

  {  process  }

  lines(4);
  writeln('      Creating TARBUL No. ',tbnum);
  lines(1);
  writeln('       **  Function Complete  **');
  lines(1);
  spacebar;
end;



procedure newTARBUL;

begin
```

```
 repeat
   clear;
   writeln('       Start a new TARBUL file');
   writeln(dots);
   lines(1);
   writeln('       The options are:');
   lines(1);
   writeln('         1.  Start a new TARBUL file');
   writeln('         2.  Information');
   writeln('         3.',return);
   lines(1);
   select;
   read(ch);
   if ch in ['1','2','3','R','r','?'] then
   begin
     if eoln(input) then readln;
     case ch of
       '1': begin
              lines(1);
              writeln('      Creating a new TARBUL file....');
              lines(2);
              tbnum := tbnum + 1 ;

              {  process  }

              writeln('    **  TARBUL file number ',tbnum,'  created  **');
              lines(2);
              spacebar;
              exit(newTARBUL);
            end;
       '2','?' : begin
                   lines(2);
writeln('    After the TARBUL is printed and published, a new TARBUL');
writeln('record must be created so that information for the next TARBUL');
writeln('can be collected. This procedure erases the old TARBUL, creates');
writeln('a new TARBUL file and automatically updates the TARBUL number.');
                   lines(1);
                   spacebar;
                 end;
       '3','R','r' : exit(newTARBUL);
     end
   end
   else menuerror;
  until menuloop = true;
 end;


procedure TBinfo;

  begin
    lines(4);
writeln('    The transactions of the targets on the target list have');
writeln('been recorded in the TARBUL file. When determined by the TIO and');
writeln('the FSC, the TARBUL in the correct message format for transmission'
writeln('can be printed from this file. It contains additions, deletions'
writeln('and changes to the target list since the last TARBUL was printed'
lines(1);
writeln('     After the TARBUL is printed, you must start a new TARBUL fil
writeln('The system will remind you of this when you print the TARBUL. Af
writeln('you set the TARBUL number the first time, the system will change
writeln('the TARBUL number for each succeeding bulletin.');
lines(1);
writeln('     Option 1 can be used to look at the current status of the');
writeln('TARBUL. Option 3 is used only once to number the first TARBUL');
writeln('issued by the FSCC. Option 6 returns you to the previous menu.')
    lines(2);
  end;



begin {tarbul}
  repeat
    clear;
    lines(1);
    writeln('                  TARGET BULLETIN');
    writeln(dots);
    lines(1);
    writeln('          The options are:');
    lines(1);
    writeln('               1.  Display the current TARBUL file');
    writeln('               2.  Print the current TARBUL file');
    writeln('               3.  Initialize the TARBUL number');
    writeln('               4.  Start a new TARBUL file');
    writeln('               5.  Information');
    writeln('               6.',return);
    lines(2);
    select;
    read(ch);
    if ch in menunum then
    begin
      if eoln(input) then readln;
      case ch of
        '1' : dispTARBUL;
        '2' : printTARBUL;
        '3' : numTARBUL;
        '4' : newTARBUL;
        '5','?' : begin
                    clear;
                    TBinfo;
                    spacebar;
                  end;
        '6','R','r' : exit(tarbul);
        '7','8' : menuerror;
      end
    end
    else menuerror;
  until menuloop = true;
end;
```

```
procedure printtgt;


procedure printmenu;

begin
  clear;
  writeln('          PRINT TARGET DATA');
  writeln(dots);
  lines(1);
  writeln('     The options are:');
  lines(1);
  writeln('          1.  Print the target list');
  writeln('          2.  Print the list of targets');
  writeln('          3.  Print all targets (active and inactive)');
  writeln('          4.  Print a target card');
  writeln('          5.  Information');
  writeln('          6.  Information on special target lists');
  writeln('          7.',return);
  lines(1);
  select;
end;


procedure printinfo1;

  begin
    lines(1);
writeln('    The procedure also prints all the information for a particular');
writeln('target in the target card format to provide a manual backup');
writeln('to the system in case of a failure. If desired, all targets');
writeln('from the data base can be printed in this format. Information');
writeln('on printing lists with special parameters can be obtained');
writeln('from option 3 of the main command menu.');
    lines(1);
  end;


procedure printinfo;

begin
  clear;
writeln('    This procedure prints the formatted target information');
writeln('to the line printer for record purposes and for target processing');
writeln('for FSCC personnel. The basic format for the list is as follows:');
lines(1);
writeln('TGT NO  CL PRI LOCATION  ALT   SAASG    DESCRIPTION');
writeln('------  -  --- --------  ----  -----    -----------');
writeln('AD2223* A    I 34567545    92   AIR     2 T-62 TANKS IN OPEN');
writeln('AD2224  D   IV 34557555  122   NGF     FORTIFIED BUNKER COMPLEX');
writeln('AD2235* B   II 34527599    52   ARTY    PIT OF T-122 AA GUNS');
writeln('NA2211* A    I 54225722    12   NFG     BEACH FORTIFICATIONS');
writeln('AB2K66  C   I.I [illegible]   125   AIR     [illegible]');
writeln('AB2011  B   IV 3-55-777     42    NONE    5 SCHOOL BUILDINGS');
lines(1);
writeln('    NOTE: * indicates target list');
  printinfo1;
  spacebar;
end;



procedure specialinfo;

begin
clear;
lines(2);
writeln('        Information on Special Target Listings');
writeln(dots);
lines(2);
writeln('      The system has the capability of providing lists of');
writeln('targets organized and sorted by parameter. These parameters');
writeln('include the following:');
lines(1);
writeln('             Priority');
writeln('             Classification');
writeln('             Type');
writeln('             Accuracy');
writeln('             Status');
writeln('             Supporting Arm Assigned');
lines(1);
writeln('     Additional information on the procedure which does this');
writeln('can be obtained by selecting option 7 of the main command menu.
writeln('To return to this menu, retrace through the previous menus by'
writeln('selecting the menu return option or by typing an R for each men
lines(1);
spacebar;
end;




procedure printTL;

begin
  clear;
  lines(4);
  write('        Printing the Target List...');
  for t := 1 to 12 do
  begin
    delay;
    write(dot);
  end;
  lines(5);
  writeln('           **  Function Complete  **');
  lines(1);
  spacebar;
end;


procedure printLOT;

begin
  clear;
  lines(4);
  write('         Printing the List of Targets...');
  for t := 1 to 12 do
  begin
    delay;
    write(dot);
  end;
  lines(5);
  writeln('          **  Function Complete  **');
  lines(1);
  spacebar;
end;



procedure printall;

begin
  clear;
  lines(4);
  write('   Printing All Active and Inactive Targets...');
  for t := 1 to 12 do
  begin
    delay;
    write(dot);
  end;
  lines(5);
  writeln('          **  Function Complete  **');
  lines(1);
  spacebar;
end;


procedure printcard;

var  tnum : string[5];

begin
  repeat
    getout := false;
    clear;
    lines(2);
    writeln('      ENTER TARGET NUMBER');
    prompt;
    readln(tnum);
    if (tnum = 'ALL') or (tnum = 'all') then lines(1
    else if length(tnum) = 3 then exit(printcard)
    else if length(tnum) <> 6 then
    begin
      getout := true;
      lines(1);
writeln(' The target number consists of two letters and four numbers.');
writeln(' For example, AB1276 or AA7701. Please reenter data.');
lines(1);
writeln(' To print all of the target cards, type ALL and press RETURN.');
      lines(1);
      returnbar;
    end;
  until getout = false;
  lines(5);
  write(' Printing the Target Card for target ',tnum,'...');
  for t := 1 to 12 do
  begin
    delay;
    write(dot);
  end;
  lines(3);
  writeln('          **  Function Complete  **');
  lines(1);
  spacebar;
end;



begin {printtgt}
  repeat
    printmenu;
    read(ch);
    if ch in menunum then
    begin
      if eoln(input) then readln;
      case ch of
        '1' : printTL;
        '2' : printLOT;
        '3' : printall;
        '4' : printcard;
        '5','?' : printinfo;
        '6' : specialinfo;
        '7','R','r' : exit(printtgt);
        '8' : menuerror;
      end
    end
    else menuerror;
  until menuloop = true;
end;


procedure stats;
```

```
begin
  clear;
  lines(4);
  write('  Processing Target Statistical Information.....');
  for t := 1 to 12 do
  begin
    delay;
    write(dot);
  end;
  lines(5);
  writeln('          ** STATISTICAL INFORMATION DISPLAY **');
  lines(4);
  returncar;
end;


procedure eraseinfo;

  begin
    lines(5);
writeln('    This procedure will erase every file of the data base. It');
writeln('will destroy the target information, TARGET file, passwords and');
writeln('the system directory. This is done primarily to remove all of');
writeln('the classified information from the diskette.');
lines(3);
writeln(' The initialization procedure will also do this. This procedure');
writeln('declassifies the diskette and should only be used at the end');
writeln('of an operation and the data is no longer needed.');
    lines(1);
  end;



procedure erase;

begin
  repeat
    clear;
    lines(1);
    writeln('            Erasing the Target File');
    writeln(dots);
    lines(1);
    writeln('            The options are:');
    lines(1);
    writeln('              1.  Information');
    writeln('              2.  Erase file');
    writeln('              3.',return);
    lines(1);
    select;
    read(ch);
    if ch in ['1','2','3','R','r','?'] then
    begin
      if eoln(input) then readln;
      case ch of
```

```
        '1','[illegible]' : begin
                     clear;
                     eraseinfo;
                     spacebar;
                   end;
        '2' : begin
                  lines(3);
                  write('      Erasing All Files.....');
                  filecheck := true;
                  initialize;
                  clear;
                  lines(5);
writeln('                   ** Function Complete  **');
lines(3);
writeln('        Diskette file erased....return to main command menu.');
lines(2);
writeln(' **  Halt operation by selecting option 7 on the main command m
writeln('                .........then restart system');
                  lines(2);
                  spacebar;
                  getout := true;
                  exit(erase);
                end;
        '3','R','r' : exit(erase);
      end
    end
    else menuerror;
  until menulcop = true;
end;



procedure copyDbinfo;

  begin
    lines(2);
writeln('   **  Place the current target diskette in disk drive A **');
writeln('                        (right side)');
lines(1);
writeln('   **  Place the back-up diskette in disk drive B  **');
writeln('                        (left side)');
lines(1);
writeln('   **  Press the RETURN key. The system will automatically');
writeln('       copy the target information from drive A to drive B.  **
lines(1);
writeln('   **  When FUNCTION COMPLETE appears, do the following:');
writeln('         * Remove the back-up diskette from drive B');
writeln('         * Remove the target diskette from drive A');
writeln('         * Place the system diskette back in drive A');
writeln('         * Place the target diskette back in drive B');
writeln('         * Press the RETURN key');
lines(2);
writeln('      PRESS RETURN TO COPY DATA BASE');
    prompt;
  end;


procedure copyDB;

var copydir : string[4];

begin
  clear;
  lines(1);
  writeln('              Copy Data Base Procedure');
  writeln(dots);
  lines(6);
writeln('   This procedure allows you to make a back-up copy of');
writeln('   the target diskette. It requires you to switch the');
writeln('   diskettes in the disk drives and use a pre-formatted');
writeln('   back-up diskette. If you do not desire to copy the');
writeln('   data base, then press the RETURN key to return to the');
writeln('   previous menu. The directions in this section must be');
writeln('   followed exactly.');
  lines(3);
  writeln('    **  Type COPY and press the RETURN key');
  prompt;
  readln(copydir);
  if (copydir = 'COPY') or (copydir = 'copy') then
  begin
    clear;
    copyinfo;
    repeat
      read(ch);
    until eoln(input);
    clear;
    lines(5);
    write('         COPYING DATA BASE...');
    for t := 1 to 10 do
    begin
      write(dot);
      delay;
    end;
    lines(5);
    writeln('     **  FUNCTION COMPLETE **');
    lines(1);
    returncar;
  end
  else exit(copyDB);
end;




 procedure utilmenu;
```

```
begin
  clear;
  lines(1);
  writeln('              THE SYSTEM UTILITY FUNCTIONS');
  writeln(dots);
  lines(1);
  writeln('          The options are:');
  lines(1);
  writeln('             1.  Change the password');
  writeln('             2.  Copy the data base file');
  writeln('             3.  Construct the TARBUL');
  writeln('             4.  Print the target data');
  writeln('             5.  Display target file statistics');
  writeln('             6.  Erase the target files');
  writeln('             7.  Information on these functions');
  writeln('             8.',return);
  lines(1);
  select;
end;


procedure utinfo;

  begin
writeln('     The fourth option prints the list of targets, the target li
writeln('and target information in the target card format. Listing of ta
writeln('by special parameters (like all Class A, priority II) is handle
writeln('by a different procedure. The statistics display shows a numeri
writeln('breakdown of the categories of information in the list of targe
lines(1);
writeln('     Option 6 erases all the information from the diskette. This
writeln('done at the end of an operation to declassify the diskette file
writeln('The last option returns you to the main command menu.');
  lines(1);
  end;


     procedure utilinfo;

    begin
      clear;
      lines(2);
writeln('     This section provides various housekeeping procedures for');
writeln('the TIO. The first option allows you to change the passwords'),
writeln('for the system users. The second permits you to copy the targe
writeln('files from the target diskette to a second diskette to functio
writeln('as a backup file. The third option constructs a target bulleti
writeln('(TARBUL) from all the data base transactions since the last TA
writeln('was printed. The routine will let you view the current TARBUL');
writeln('information and print it in the proper message format.');
      lines(1);
      utinfo;
    spacebar;
  end;



begin {utility}
  menunum := ['1','2','3','4','5','6','7','8','Q','q','?'];
  clear;
  getfile('notimp.text');
  spacebar;
  repeat
    utilmenu;
    read(ch);
    if ch in menunum then
    begin
      if eoln(input) then readln;
      case ch of
        '1' : changeFN;
        '2' : copyFN;
        '3' : tarbul;
        '4' : printtxt;
        '5' : stats;
        '6' : begin
                erase;
                if getout then
                begin
                  getout := false;
                  exit(utility);
                end;
              end;
        '7','?' : utilinfo;
        '8','Q','q' : exit(utility);
      end
    end
    else menuerror;
  until menuloop = true;
end;
```

:::::::::::::::::::::: DELTGT.TEXT :::::::::::::::::::::::::::

This procedure will erase the target from the list of targets. Targets are usually deleted when they have been put in the dead file, that is, they are destroyed, overrun by friendly forces or inactive for a long period of time. A printed copy of the target should be retained in case it must be reactivated. To do this, select the utility function from the main command menu.

::::::::::::::::::::::: DISPTGT.TEXT ::::::::::::::::::::::::::

This procedure selects a target from the list of targets and displays it on the screen in the target card format. Up to three target surveillances can be included if the target has been attacked. This display format can be printed by using the print procedure in the utility function option from the main command menu.

:::::::::::::::::::::: DISTGTMENU.TEXT ::::::::::::::::::::::::

The target can be found by using the target number or the grid coordinates. Select the appropriate choice from this menu.

1. Target number
2. Grid coordinates

::::::::::::::::::::::: PROCINFO.TEXT :::::::::::::::::::::::::

These options allow you to enter the target into the data base file, display it on the screen, make changes to the target information or discard all of the input. Selection of any of these options will return you to the previous menu after processing is complete.

```
:::::::::::::::::::::::: ADDTGTINFO.TEXT ::::::::::::::::::::::::

    This procedure allows you to construct a new target report.
It will prompt you for 22 different items of target information
for each target. Some items are mandatory and you must enter the
appropriate information. These mandatory items are:


           a.  Target number
           b.  Grid location
           c.  Description
           d.  Priority
           e.  Classification
           f.  Status
           g.  Target type


    Other items like map reference or remarks can be skipped or
left blank by merely pressing the RETURN key.

:::::::::::::::::::::::: FUNIT1.TEXT ::::::::::::::::::::::::::::

    The firing unit is the designation of the supporting arms
unit which undertakes the attack on the target. The input must
not exceed 6 letters or characters. An example of a firing unit
would be, Air...2 A-4, Artillery...F-1-12, Naval Gunfire...DDG-4.
Enter the appropriate unit and press the return key.

:::::::::::::::::::::::: ROUNDS.TEXT ::::::::::::::::::::::::::::

    Enter the number of rounds, bombs or ordnance used on the
target and indicate the type or caliber of the ordnance. Limit
the input to 12 characters. Use a slash or a dash to separate
the number and type if needed. If the information is not known,
press the return key. An example of input would be: Air..2 / B-6,
Artillery...36 / 8in HE, NGF...12 - 5in 54.

:::::::::::::::::::::::::: FUNIT2.TEXT ::::::::::::::::::::::::::

     The length of the input exceeds six characters.
     please reenter the firing unit in six characters
     or less and press the RETURN key.

:::::::::::::::::::::::: DAMAGEMEN.TEXT :::::::::::::::::::::::

          The options are:

                1.  Damaged
                2.  Destroyed
                3.  Interdicted
                4.  Harassed
                5.  Neutralized
                6.  Illuminated
                7.  Unobserved
```

```
            e.  Unknown


::::::::::::::::::::::::: ROUNDS1.TEXT :::::::::::::::::::::::::


          The length of the input exceeds ten characters.
          Please reenter the rounds fired in ten characters
          or less and press the RETURN key.

::::::::::::::::::::::::: DAMREP.TEXT :::::::::::::::::::::::::

     Enter the reported damage assessment by selecting one
of the above 8 choices. This is the information that is reported
by the observer or intelligence source which attacked or observed
the target.

::::::::::::::::::::::::: DAMASS.TEXT :::::::::::::::::::::::::

     Enter the estimate of the actual or suspected damage to the
target based on the reported surveillance, the ordnance used and
supporting arm employed and other intelligence reports received.
It does not necessarily have to agree with the damage reported.

::::::::::::::::::::::::: MANDMSG.TEXT :::::::::::::::::::::::::

    ** Input of this item is required by the target file.
    Processing cannot continue without inputting the appropriate
                  data. Please reenter the data.

       For information, type a ? after the entry prompt.

::::::::::::::::::::::::: TNUM.TEXT :::::::::::::::::::::::::

    The target number consists of two letters and four numbers.
An example of a correct entry would be [illegible]. Use upper case
letters for the target designator. Please reenter the data.

::::::::::::::::::::::::: TGTLOC.TEXT :::::::::::::::::::::::::

     The grid coordinates of the target should be 8 numbers long
and consist only of numbers. For example, [illegible] or [illegible].
A six number coordinate such as 345678 can be changed to an eight
digit one by adding zeros to give 34056780. Please reenter data.

::::::::::::::::::::::::: TGTDESC.TEXT :::::::::::::::::::::::::

     The target description can be up to 30 characters long and
include both numbers and letters. If the description exceeds 30
characters, then all characters passed 30 will be discarded. Be
sure to indicate the quantity and the type of target. For example.
5 ZSU 23-4 in open field .
```

```
:::::::::::::::::::::::: CLASSMENU.TEXT ::::::::::::::::::::::

      ENTER TARGET CLASSIFICATION

       The options are:

          1.  A
          2.  B
          3.  C
          4.  D
          5.  E

:::::::::::::::::::::::: TGTPRIMENU.TEXT :::::::::::::::::::::

        ENTER TARGET PRIORITY

         The options are:

            1.  I
            2.  II
            3.  III
            4.  IV

:::::::::::::::::::::::: TYPEMENU.TEXT :::::::::::::::::::::::

      ENTER TARGET TYPE

       The options are:

          1.  Tank
          2.  SEAD target
          3.  Installation
          4.  Counter Battery
          5.  Observation Post
          6.  Terrain
          7.  Vehicles
          8.  Fortifications
          9.  Miscellaneous

:::::::::::::::::::: TGTTYPE.TEXT ::::::::::::::::::::::::::

    The target type is determined from the target description. This
information will be used to group targets of the same or similar
type together. If the target cannot be classified into the above
categories, use miscellaneous (option 9).

:::::::::::::::::::: TGTALT.TEXT :::::::::::::::::::::::::::

    The altitude is entered in meters and consists of up to
four digits from 0000 to 9999. For example, 432. The altitude
must be entered in meters. If the altitude is not known, then
press the return key. Please reenter data.

:::::::::::::::::::::: SA.TEXT :::::::::::::::::::::::::::::
```

The target can be assigned to tactical air, naval gunfire or artillery or any combination of supporting arms; it can also be assigned to another supporting arm such as tanks in which case option 5 should be chosen. This should be noted in the remarks section. If a supporting arm is not assigned, select option 6. If this section is skipped, option 6 is automatically assigned.

:::::::::::::::::::::::: REMARKS.TEXT ::::::::::::::::::::::::::

The remarks section can be up to 40 characters long and include both numbers and letters. If the remarks exceed 40 characters, then all characters passed 40 will be discarded. Remarks usually include recommendations of attack mode, schedules of fire, attack restrictions and other important information. For example...attack with main armament, HE Q and VT, area fire at H-20.

:::::::::::::::::::::::: MAPSHT.TEXT ::::::::::::::::::::::::::

Indicate the map and sheet number but do not exceed 10 characters. This entry can be detailed '3400 IV 1:A 501' or be informal (ie China) as determined by the TIO.

:::::::::::::::::::::::: SOUR.TEXT ::::::::::::::::::::::::::

Indicate from what source of intelligence the target was obtained; intelligence summary, air observer, forward observer, aerial photo, reconnaissance sensor or other source. For example, INTSUM 3222 121400Z JAN 81. Do not exceed 20 characters.

:::::::::::::::::::::::: AFOTONUM.TEXT ::::::::::::::::::::::::::

If the target was obtained from an aerial photograph enter the photo number and press the return key. For example, A2111234. If the target came from another source then just press the return key.

:::::::::::::::::::::::: PHOTOGRID1.TEXT ::::::::::::::::::::::::::

** If the photo grid location is the same as the target location, enter an 0 and press the return key. Otherwise, enter an eight digit number and press the return key.

ENTER PHOTO COORDINATES

:::::::::::::::::::::::: PHOTOGRID2.TEXT ::::::::::::::::::::::::::

The coordinates of the photo target should be 8 numbers long and consist only of numbers. For example, 59660642 or 67824078. A six number coordinate such as 345678 can be changed to an eight digit one by adding zeros to give 34506780.

If the photo grid location is the same as the target location then, enter an S and press the return key. Please reenter data.

:::::::::::::::::::::: TGTACC.TEXT ::::::::::::::::::::::::::

The 4 choices represent the accuracy of the target information. A target which is known to exist is a confirmed target. A probable and a possible target are suspected to exist. If the intelligence evaluation does not fall into one of these 3 areas, then enter option 4 for unknown. If this item is skipped, the system will automatically enter unknown in the target record.

:::::::::::::::::::::: CHANGINFO.TEXT ::::::::::::::::::::::

This procedure can only be used on targets which have already been created and exist in the list of targets. You must page your way through the existing target information until you reach the item you want to change. The system will display the current information and ask if you desire to change.

A YES response will cause the item to be changed. A NO or a RETURN key will cause the system to go on to the next item. The target information will not be changed unless you press the Y key. You can leave the procedure at any time by typing a Q.

Changes to BDA or additions to the BDA portion of the target record are performed by option 2 of the target menu. Enter an P after returning to the menu.

:::::::::::::::::::::: CHANGEM.TEXT ::::::::::::::::::::::::

The first option will display the target card. The second checks each item of target information and allows you to change those areas desired. The third option writes the changes to the file and the last option returns you to the previous menu.

:::::::::::::::::::::: PRIORITY.TEXT ::::::::::::::::::::::::

Target Priorities

....

PRIORITY I....Targets capable of preventing the execution
              of the plan of action by the landing force
              and its elements.

PRIORITY II...Targets capable of immediate serious interference
              with the plan of action by the landing force
              and its elements.

PRIORITY III..Targets capable of ultimate interference with
              the plan of action by the landing force
              and its elements.

PRIORITY IV...Targets capable of limited interference with

the plan of action by the landing force
and its elements.

:::::::::::::::::::::: CKDONEX.TEXT :::::::::::::::::::::::::::

Option 1 will return you to those items which have not been completed so that you may enter the necessary data. Option 2 allows you to proceed to the processing menu where you can put the target in the file, display it on the screen, make changes to the data or delete the target completely. Option 3 allows you to select the change option from the processing menu.

:::::::::::::::::::::: ADDTNG1.TEXT :::::::::::::::::::::::::::

You may correct information entered by using the RUBOUT key, but after you have pressed the return key, the only way to change the data is by the Change Target procedure.

This procedure prompts in sequence until all items are entered. You will then be given the option of writing the new target to the file, displaying the information on the screen, changing some of the entries or discarding the information just entered.

To end this procedure at any time, type a Q and a RETURN key.

To obtain help in inputting the correct data, type a ? when the system prompt asks for the target data. Specific information and an example of the correct entry will be displayed.

Follow all data entries with the RETURN key.

:::::::::::::::::::::: MMINFO1.TEXT :::::::::::::::::::::::::::

This is the main command menu of the system

Option 1...provides information on how to operate the system, doctrinal guidelines for target information, security requirements, formats used, target analysis guidelines and other items.

Option 2...enables you to add a target to or delete a target from the target file, change information about a target, and display all the information for a particular target on the screen

Option 3...enables you to obtain a target list by specifying a parameter or a list of parameters. These parameters include classification, priority, status (active/inactive/attacked), and supporting arm assigned. All information for a particular target can also be displayed.

```
::::::::::::::::::::: MAINFO2.TEXT :::::::::::::::::::::::

   Option 4...allows you to change the password, copy the
              data base to another diskette, erase the data
              base, print target lists and target cards, display
              and print the TARSUL as well as provide a statistical
              breakdown of the list of targets.

   Option 5...initializes the target information system for a
              new operation. All files will be re-created so that
              new information can be entered. The current target
              files will be discontinued.

   Option 6...provides this information about the system. More
              detailed information can be obtained from Option 1.

   Option 7...halts the operation of the system after writing
              important information to the diskette files. The
              system will have to be restarted to use the target
              file.


:::::::::::::::::::::::: MAINFO6.TEXT :::::::::::::::::::::::

          If you need help at any time, type a [illegible].



   If you  want to  return to  the previous menu enter the
    option number provided by the current menu or type an R.

:::::::::::::::::::::::: TGTMODM.TEXT ::::::::::::::::::::::::

     These procedures operate on the main list of targets. Option 1
allows you to add a new target to the list of targets. Option 2
allows you to change any information about a target currently in
the target file. Option 3 displays all the information about a
particular target on the screen in a target card format. The
target can be found either by target number or grid coordinates.


     Option 4 allows you to add a new target surveillance or
BDA to the target based on the results of attack by supporting
arms. Option 5 deletes a target completely from the list of
targets. The final option returns you to the main command menu.
This menu also allows you to use the letter A for adding targets.
D for displaying targets and C for changing targets.

::::::::::::::::::::: DTGOFBDA.TEXT :::::::::::::::::::::::

     Enter the date-time-group that the target was attacked by
supporting arms. The first 2 digits are the day of the current month
```

(1..31) and the next 4 are the local time (0001..2359).
The letter indicates the time zone. For example, 1632 on 21 May
in the ROMEO time zone would be written, 211632R. Enter the
data and press the return key. If the DTG of activation is
not known, then just press the return key.

:::::::::::::::::::::: BDASRV.TEXT ::::::::::::::::::::::::

The Battle Damage Assessment (BDA) can be up to 40
characters long. This is a concise narrative of the surveillance
of the target based on the observer report. If there is no report
or observation, then this section can be skipped. An example of a
BDA is: 4 secondary explosions, 3 vehicles burning.

:::::::::::::::::::::: BDAINFO.TEXT ::::::::::::::::::::::::::::

Each target has three spaces for recording the target
surveillance as a result of attack by supporting arms. This
procedure prompts you for each piece of information for the
Battle Damage Assessment (BDA). If a fourth BDA is entered,
it will write over the first BDA since that is the oldest.

Information on multiple surveillances can be included in the
remarks section of the target record by using the change target
procedure. Use the display option to view the current BDA recorded
in the target file.

:::::::::::::::::::::: DTGACT.TEXT :::::::::::::::::::::::::

Enter the date-time-group that the target was activated
or added to the list of targets. The first 2 digits are the day
of the current month (1..31) and the next 4 are the
time (0001..2359). The letter indicates the time zone.
For example, 1632 on 21 May in the ROMEO time zone would
be written, 211632R. Enter the data and press the return key.
If the DTG of activation is not known, then just press the
return key.

:::::::::::::::::::::: CLASS.TEXT :::::::::::::::::::::::::

Target Classification

....

CLASS A....Targets that threaten ships, aircraft, minesweeping
and UDT operations.

CLASS B....Targets that threaten assault forces in the ship-
to-shore movement and assault of the beach.

```
CLASS C....Targets that threaten or oppose landing force
           operations after landing or affect the ability
           of the enemy to continue resistance.

CLASS D....Targets that will not be fired upon prior to
           D-Day.

CLASS E....Targets that must not be destroyed because of
           probable future use or humanitarian reasons unless
           authorized by the commander.
 .................................................................


::::::::::::::::::: TGTACCMENU.TEXT :::::::::::::::::::::::::::::

          ENTER TARGET ACCURACY

           The options are:

              1.  Confirmed
              2.  Probable
              3.  Possible
              4.  Unknown

:::::::::::::::::::: SAMENU.TEXT :::::::::::::::::::::::::::::::

      ENTER SUPPORTING ARM ASSIGNED TO TARGET

          The options are:

                 1.  ARTY
                 2.  NGF
                 3.  AIR
                 4.  AIR, ARTY
                 5.  AIR, NGF
                 6.  ARTY, NGF
                 7.  AIR, ARTY, NGF
                 8.  Other
                 9.  None

:::::::::::::::::::: CHANGEMENU.TEXT ::::::::::::::::::::::::::

         The options are:

              1.  Display the target card
              2.  Page through the target record
              3.  Write the change to the file
              4.  Return to previous menu

::::::::::::::::::::: PRETGT.TEXT ::::::::::::::::::::::::::::

              INSTRUCTIONS FOR ADDING A TARGET
 .................................................................
```

```
            The system will ask for each item of information.

        Enter the required information and press the RETURN key.

To leave this procedure at any time, type a Q after the entry prompt

 To skip a section, just press a RETURN key and the system will go
 to the next item of information unless  the information requested
      is mandatory. In that case you must enter information.

    To receive more information about this procedure, type a ?

          **     To continue, press the RETURN key      **

:::::::::::::::::::::::::::::::::::::::::::::::.:::.:::::::.:::
```

DISTRIBUTION

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93940	1
4.	Professor Lyle A. Cox, Jr., Code 52Cl Department of Computer Science Naval Postgraduate School Monterey, California 93940	1
5.	Commanding General Attn: Fire Support Coordinator 1st Marine Division, FMF Camp Pendleton, California 92055	1
6.	Commanding General Attn: Fire Support Coordinator 2nd Marine Division, FMF Camp Lejeune, North Carolina 28542	1
5.	Commanding General Attn: Fire Support Coordinator 3rd Marine Division, FMF FPO San Francisco 96602	1
7.	Commanding General Education Center Attn: Supporting Arms Branch MCDEC Quantico, Virginia 22134	1

8. Commanding General Development Center MCDEC Quantico, Virginia 22134	1
9. Commanding General Attn: Fire Support Section (TEC) MCAGCTC Twentynine Palms, California 92227	1
10. Marine Corps Representative U. S. Army Artillery School Fort Sill, Oklahoma 73503	1
11. TRADOC Research Element Monterey, Code TREM Naval Postgraduate School Monterey, California 93940	1
12. Commanding Officer MCTSSA Attn: MIFASS Team Camp Pendleton, California 92055	1
13. LtCol Ronald J. Coulter, USMC 6033 Burnside Landing Drive Burke, Virginia 22015	1