DESIGN AND DEVELOPMENT OF AN AUTOMATED BANK TELLER

by

PURNENDU SINHA

UNDER THE SUPERVISION OF

DR. B.P. SINHA

B P Sinha
4/8/88

ACKNOWLEDGEMENT

I take this opportunity to thank Dr. B.P.Sinha for his invaluable help without which this dissertation would never have been completed successfully.

I would also like to thank Mr. S.K.Badruzzaman for his kind help and encouragement throughout the dissertation.

Finally, thanks to Mr. George M. George for clearing my doubts at different stages of dissertation.

CONTENTS

Introduction

Chapter 1 ANALYSIS

1.1 Problem Definition

1.2 Requirement Specification

Chapter 2 DESIGN

2.1 Dataflow Diagrams

2.2 Hierarchical Structure of different modules

2.3 Module Specification

2.4 Details of each module

2.41 User module

2.42 Manager module

2.5 Details of file structures.

Chapter 3 USER MANUAL

Chapter 4 REFERENCES.

INTRODUCTION

One of the most important functions of modern bank management is the introduction of mechanisation in different areas of banking. Most of the operations carried out in banks are repetitive in nature and consequently, a substantial amount of time can be saved with proper mechanisation. Naturally, therefore a bank with proper automation can provide much better facilities and service to its customers which is of utmost importance, in view of the cut-throat competition between the various commercial banks. With this background in view, this dissertation was taken up to automatise one vital service area of the banking system, namely, the teller service. It was also proposed to offer a few more services connected to the management of bank as a whole.

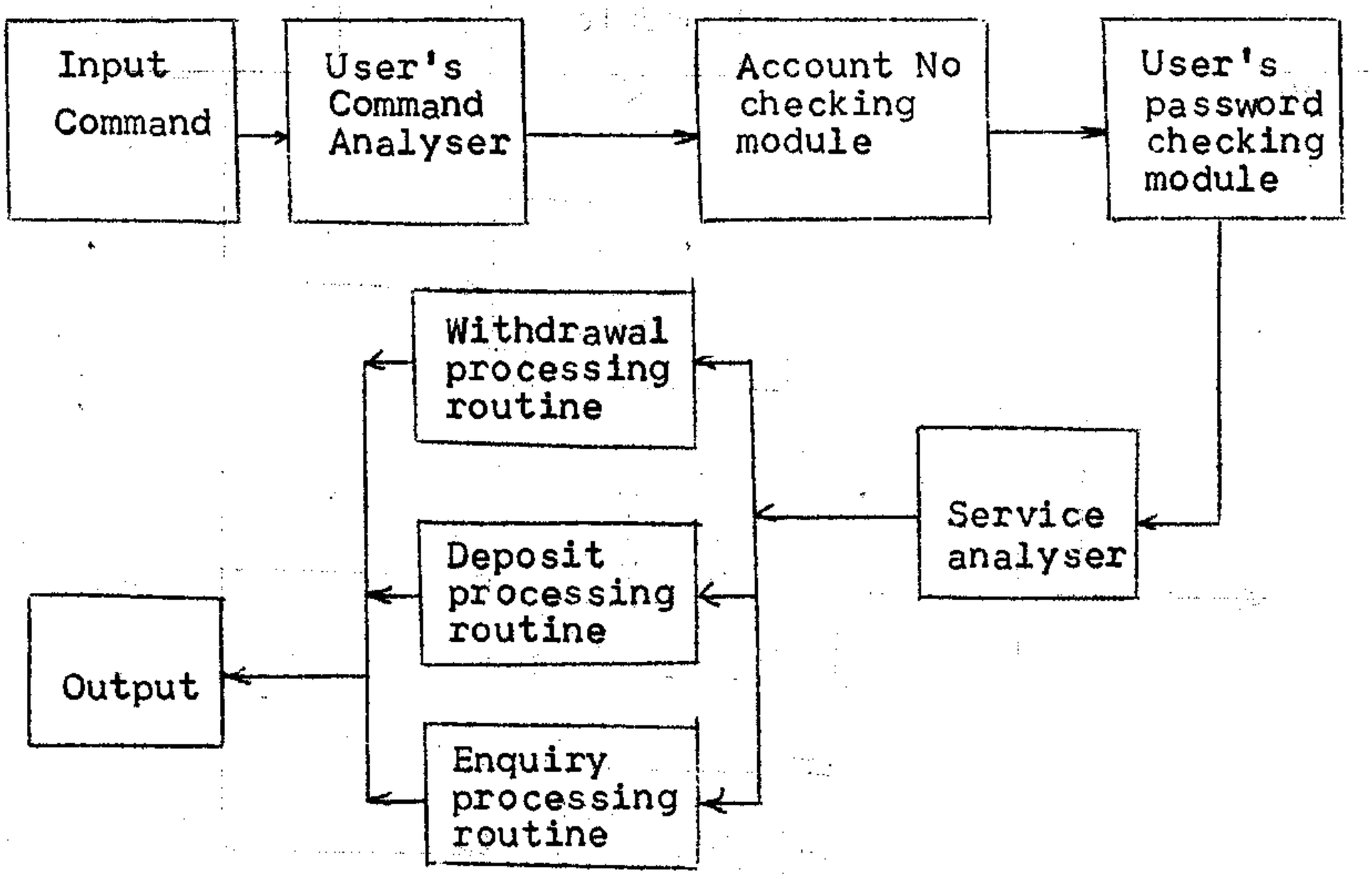
Language used for implementation : TURBO PASCAL

Time limit : 10th May 1988 to 10th July 1988.

CHAPTER 2.1 :

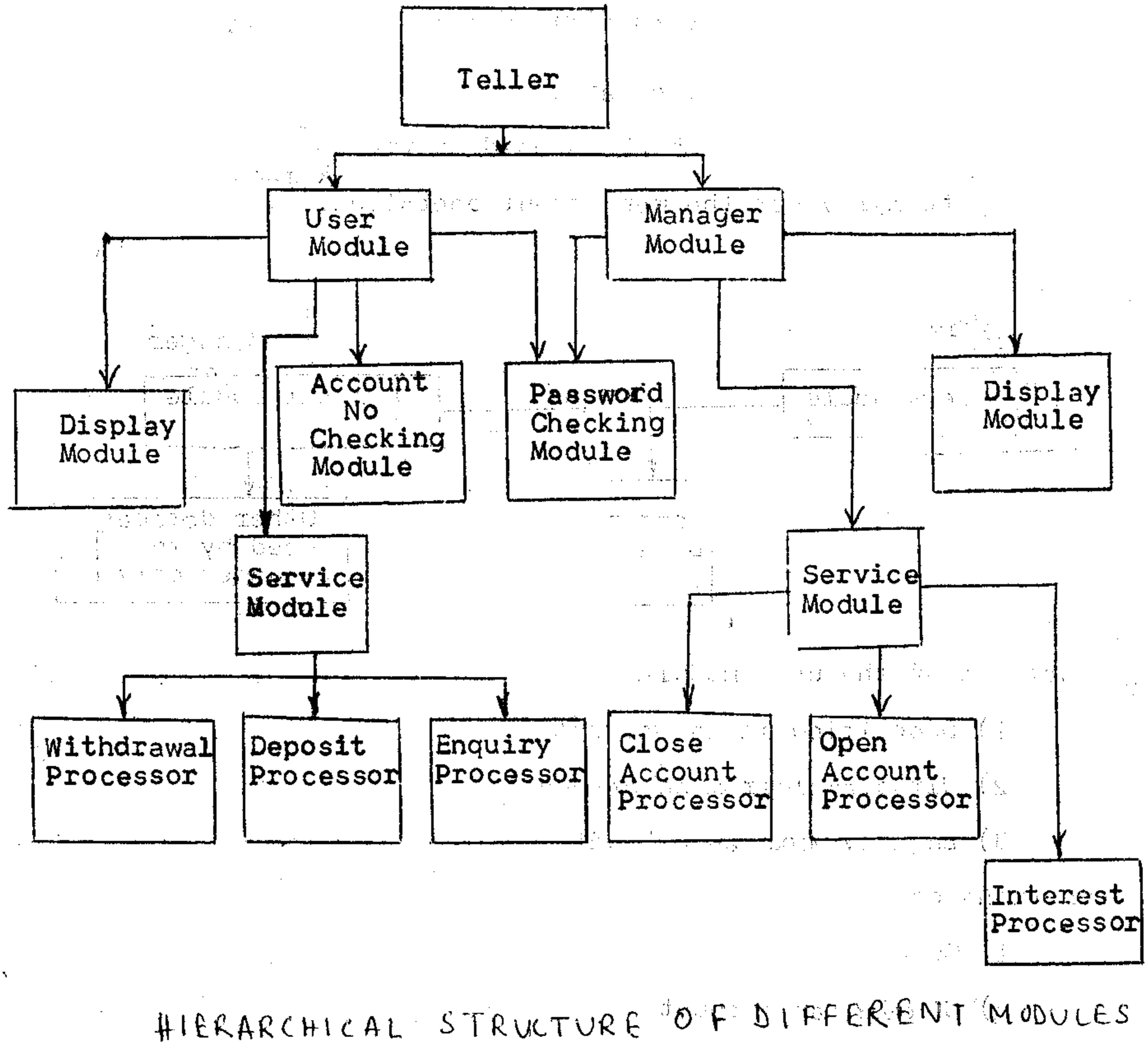
DATAFLOW DIAGRAMS

User's Section



Contd.....

Chapter 2.2 :



HIERARCHICAL STRUCTURE OF DIFFERENT MODULES

Contd.....

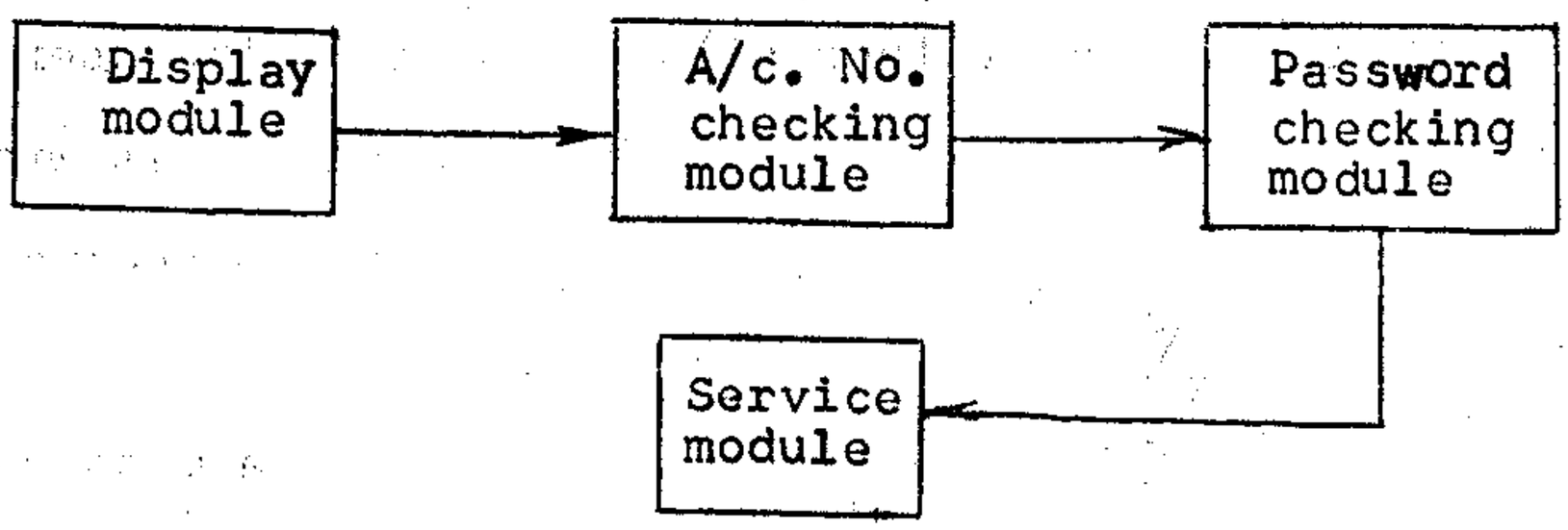
Chapter 2.4 :

Details of each module

2.41 User module :

This module is further divided into several submodules :

1) Display module
2) Account no checking module
3) Password checking module
4) Service module



Functions of different submodules :

1) Display module : This module takes care of all the interacting between the user and the teller.
2) A/c. no checking module : This module checks the account no specified the user.
3) Password checking module : This module checks the users password.
4) Service module : The module serves the customer according to their reque:

Contd.....

The action routines under service module.

1) Withdrawal processing routine

2) Deposit processing routine

3) Enquiry processing routine

Details of each action routine and their input and output parameters.

1) Withdrawal Processing Routine :

Inputs : 1) account type

2) account number

3) amount of money to be withdrawan

4) the type of currency notes to be given to the user

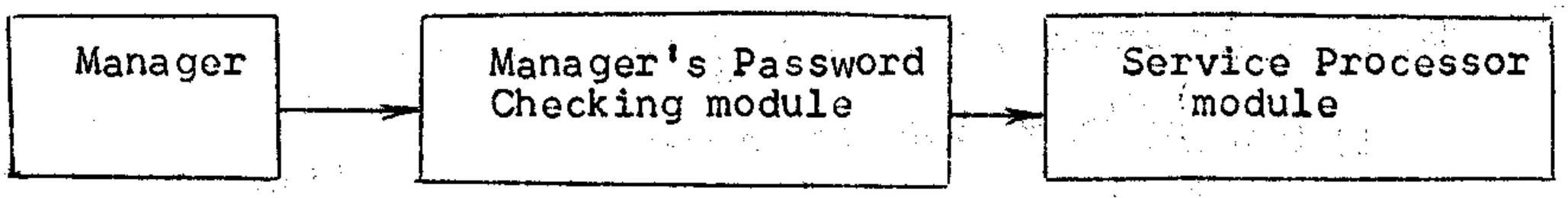
5) date of transaction supplied by the system.

Outputs : any one of the following messages.

1) the user does not have enough money and his current account saving is such and such

2) the service cannot be provided because already one withdrawal transaction has taken place on that day.

Contd.....



The functions of the above two modules

1) Manager's Password Checking module : If checks the password of the manager.

2) Service Processor module : If processes the service requested by the manager.

Details of each module inputs and outputs :

Manager's Password Checking module :

Inputs : Password given by the manager

Outputs : Either 1) a message that the password is wrong

or 2) a signal to the service processor module.

Service Processor module :

Inputs : type of service requested by the manager

Outputs : a call to specific action routine

Contd.....

Closing an account takes help from the following two routines

1) Account checking routine of the user module
2) Interest calculation routine
3) Password checking routine of the user module

Details of Interest Calculation routine :

This routine is divided into two parts :

1) For abnormal calculation of interest i.e. while closing the account
2) For normal calculation of interest i.e. every six months.

For Case 1.

Inputs : 1) Account No.
2) Date supplied by the system

For Case 2.

Inputs : 1) Account No.

For both the cases the outputs are same.

Outputs : 1) The details of the current Savings Account.

Contd.....

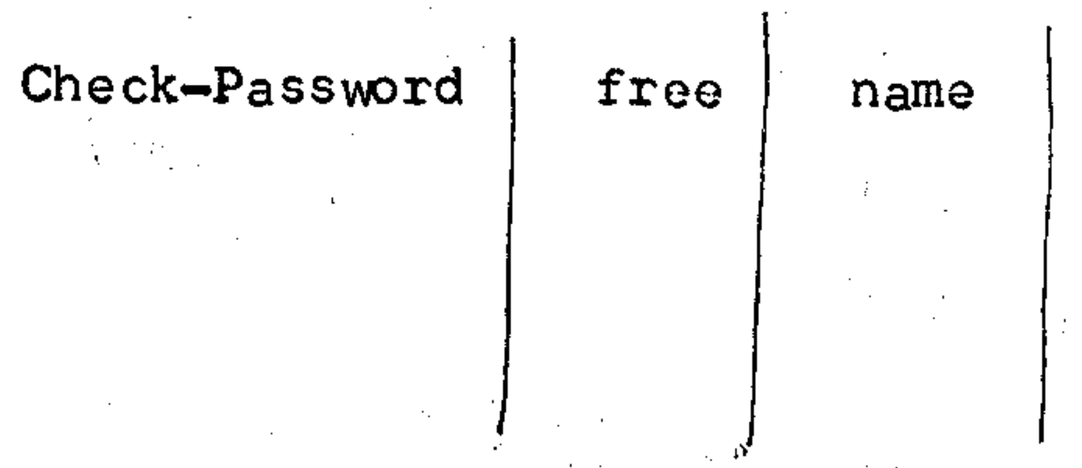
The details of the files maintained and details of their structures.

<u>The master file for each type of account</u>.

It contains all the relevant information about an account holder.

Since the no. of accounts for a particular type of account may be huge, the master file is broken into several smaller files having a unique file identifier no. we will these smaller files as subfiles.

Every subfile is of constant size and it consists of a fixed no. of sections of fixed size. Initially, only one subfile will be created and it is given as file identifier no. The information about the account holder can be obtained directly from this file by accessing the record specified by the record no field of the account no. Each record of the subfiles has the following structure.



" Check-Password" field gives the Password no for checking.

Contd.....

Account file for each account holder.

Each account holder is given an unique file which is identified by the account no. It stores the details of each transaction made to that account.

It has the follOwing structure.

Date of transaction	type of transaction	amount	total

" Date of transaction" field gives the date on which the transaction has been made.

" Type of transaction" field gives the type of transaction (withdrawal, Deposit or Interest)

" Amount" field gives the amount of money involved in transaction.

" Total" field tells us how much money is left on that account after the transaction.

Contd.....

USER MANUAL

The operations that are being carried out by the teller is menudriven. Users and the bank manager have to select the proper menu for their required service. Menus are displayed in the vedeo terminal in bright characters so that the users can easily make use of them.

For example - suppose an user wants to withdraw some money from his Savings bank account.

The following lines will be displayed first.

```
Savings  :
Current  :
```

The user will have to bring the cursor into correct row and then press ENTER.

Contd.....

REFERENCE

1) Turbo Pascal Reference Manual version 3.0

2) MSDOS Reference Manual version 3.2

3) Software Development Project Planning and Management 1982.

Bruce Phillip and Pederson S.M.

```
Program User_module;

(* This Program takes care of all the usual Bank Teller operations. *)

(* The following compiler directives are used. *)

{$c-}   {break checking of}

{$i-}   {ioresult checking of}

{$r+}   {range checking on}

{ Constant Declaration part for the Main Program }

const
     user_password_file_name              = 'user.pas';
     withdrawal                           = 'W';
     deposit                              = 'D';
     enquiry                              = 'E';
     no_of_sec                            = 99;
     sec_len                              = 10;
     max_no_files                         = 99;
     small                                = 1;
     medium                               = 2;
     medium1                              = 3;
     big                                  = 4;
     ver                                  = 5;
     hor                                  = 6;
     left_cor                             = 7;
     verybig                              = 8;
     savings                              = 'S';
     current                              = 'C';
     minamt                               = 25;


{ Type Declaration part for the Main Program  }

type
    string3                               = string[3];
    string4                               = string[4];
    string5                               = string[5];
    string6                               = string[6];
    string7                               = string[7];
    string8                               = string[8];
    string20                              = string[20];
    string40                              = string[40];
    entype                                = 0..3;
    master_rec_type                       = record
                                              date_of_op:string6;
                                              pass_field:integer;
                                              free      :char;
                                              name      :string20;
                                              addr      :string40
                                            end;
    ac_rec_type                           = record
                                              date      :string6;
                                              trans_type:char;
                                              amt,total :real
                                            end;

    stat_rec_type                         = record
                                              file_id,
```

```
                                        sec_id ,
                                        no_of_rec :integer
                                    end;
    bank_rec_type                 = record
                                        sl_no        :integer;
                                        date         :string6;
                                        trans_type   :char;
                                        amt,net_cash:real
                                    end;
```

```
( Variable Declaration part for the Main Program )

var
   out                                          :text;
   ac_no                                        :string8;
   user_password,
   file_no,
   error,
   act_ac_no,
   sec_no,
   check_password,
   type_of_currency,new_sl_no,b,n               :integer;

   user_password_file                           :file of integer;
   service_type,
   type_of_ac,
   ch                                           :char;

   user_name                                    :string20;
   user_addr                                    :string40;
   money,ptotal,ntotal                          :real;
   todate                                       :string6;

   master_rec                                   :master_rec_type;
   master_file                                  :file of master_rec_type;
   stat_rec                                     :stat_rec_type;
   stat_file                                    :file of stat_rec_type;
   ac_rec                                       :ac_rec_type;
   ac_file                                      :file of ac_rec_type;
   bank_rec                                     :bank_rec_type;
   bank_file                                    :file of bank_rec_type;
   enquiry_type                                 :entype;
   ac_file_name                                 :string7;
   ac_file_size                                 :integer;
   master_file_name,s                           :string8;
```

```
Function change(s:string6):string8;
var
   a:string8;
begin
     a[0]:=chr(8);
     a:=copy(s,1,2)+'-'+copy(s,3,2)+'-'+copy(s,5,2);
     change:=a
end;
function map(ch:char):string40;
begin
     case ch of
     'D':map:='Deposit';
     'W':map:='Withdrawal';
     'I':map:='Interest updation'
     end
end;


Procedure Clear;
```

```
(* This procedure clears the entire screen. *)

begin
     window(1,1,80,25);
     clrscr
end; (clear)

procedure wait;
var
   ch:char;
begin
     repeat
          read(kbd,ch)
     until ord(ch)=13

end;(wait)

Procedure Create_window(i:integer);

(* This procedure creates window putting bright borders around it *)

var
   ii,jj,ip,jp:integer;
 Procedure box(left,top,right,bottom:integer);
 var
    x,y:integer;
    procedure twochar(x1,y1,x2,y2:integer);
    begin
         gotoxy(x1,y1);
         write(chr(176));
         gotoxy(x2,y2);
         write(chr(176))
    end;(twochar)
 begin
      for x:=left to  right do
          twochar(x,top,x,bottom);
      for y:=top to bottom do
        begin
          twochar(left,y,right,y);
          twochar(left+1,y,right-1,y)
        end
 end;(box)
begin
     case i of
          small:begin
                     ii:=5;
                     jj:=2;
                     ip:=51;
                     jp:=15
          end;
medium:begin
                ii:=5;
                jj:=2;
                ip:=68;
                jp:=15
           end;
medium1:begin
                ii:=14;
                jj:=5;
                ip:=68;
                jp:=20
          end;
 big:begin
           ii:=4;
           jj:=1;
           ip:=80;
           jp:=24
      end;

  ver :begin
            ii:=51;
            jj:=2;
```

```
                    ip:=80;
                    jp:=15
                end;
    hor:begin

                    ii:=5;
                    jj:=15;
                    ip:=80;
                    jp:=24
            end;
    left_cor:begin
                    ii:=1;
                    jj:=1;
                    ip:=30;
                    jp:=5
                end;
    verybig:begin
                    ii:= 1;
                    jj:= 1;
                    ip:= 80;
                    jp:=25
            end

end;
    window(1,1,80,25);
    window(ii,jj,ip,jp);
    box(2,2,ip-ii,jp-jj);
    window(ii+4,jj+2,ip-4,jp-2);
    clrscr;
end;(......End of Procedure Create_window........)


procedure Display_main;

(* This procedure gives the starting display. *)

begin
     clrscr;
     create_window(verybig);
     gotoxy(2,20);
     write('Press RETURN to Start.');
     create_window(medium1);
     gotoxy(7,3);
     write('AUTOMATED BANK TELLER SYSTEM');
     GOTOXY(10,5);
     WRITE('DESIGNED AND DEVELOPED');
     gotoxy(18,7);
     write('BY');
     gotoxy(12,9);
     write('PURNENDU SINHA');

end;(.....Display_main......)

Procedure Goto_window(i:integer);
var
   ii,jj,ip,jp:integer;
begin
      case i of
          small:begin
                      ii:=5;
                      jj:=2;
                      ip:=51;
                      jp:=15
                end;
          medium:begin
                       ii:=5;
                       jj:=2;
                       ip:=68;
                       jp:=15
                  end;
           medium1:begin
```

```
                    ii:=14;
                    jj:=5;
                    ip:=68;
                    jp:=20
                end;
        big:begin
                    ii:=4;
                    jj:=1;
                    ip:=80;
                    jp:=24
                end;

         ver :begin
                    ii:=51;
                    jj:=2;
                    ip:=80;
                    jp:=15
                end;
    hor:begin

                    ii:=5;
                    jj:=15;
                    ip:=80;
                    jp:=24
                end;
            left_cor:begin
                        ii:=1;
                        jj:=1;
                        ip:=30;
                        jp:=5
                    end;
            verybig:begin
                        ii:= 1;
                        jj:= 1;
                        ip:= 80;
                        jp:=25
                    end

        end;
        window(ii+4,jj+2,ip-4,jp-2);
        clrscr
end;



Procedure cursor_movement(limit_up,limit_down:integer);
(* This procedure moves the cursor in a vertical line . *)
var
   ch       :char;
   done     :boolean;
begin
     done:=false;
     repeat
           read(kbd,ch);
           if ord(ch)=27
              then begin
                        read(kbd,ch);
                        case ch of
                             'P':begin
                                      if wherey=limit_down
                                         then write(#7)
                                       . else gotoxy(wherex,wherey+2)
                                 end;
                             'H':begin
                                      if wherey=limit_up
                                         then write(#7)
                                         else gotoxy(wherex,wherey-2)
                                 end
                            else write(#7)
```

```
                        end
                    end
                else if ord(ch)=13
                        then done:=true
                        else write(#7)
    until done;
end;(......End of Procedure cursor_movement........)


(* Procedure (2) Enter_ac_no *)
Procedure Enter_ac_no;

(* This Procedure asks for account nummber . *)
var
   done     :boolean;
   ch       :char;
begin
     goto_window(hor);
     gotoxy(3,3);
     write('Bring the cursor in the correct row and press RETURN.');
     goto_window(small);
     gotoxy(2,2);
     write('Type of Accounts');
     gotoxy(4,4);
     write('Savings.....:');
     gotoxy(4,6);
     write('Current.....:');
     gotoxy(18,4);
     cursor_movement(4,6);
     case wherey of
     4:type_of_ac:=savings;
     6:type_of_ac:=current
     end;
     goto_window(hor);
     gotoxy(3,3);
     write('Enter your Account number.');
     write('Press RETURN');
     goto_window(small);
     gotoxy(4,4);
     write(' Account No :-');

     read(ac_no);
     clrscr
end;(*..... End of procedure Enter_ac_no..... *)

(* Procedure (3) Enter_user_password *)
Function Enter_user_password:boolean;

(* This Function asks for password from the user.*)


var
   s:string5;
begin
     Enter_user_password:=true;
     goto_window(hor);
     gotoxy(3,3);
     write('Put Your Password Card Inside. ');
     gotoxy(3,4);
     write('and Press RETURN');
     wait;
     clrscr;
     assign(user_password_file,user_password_file_name);
     reset(user_password_file);
     if ioresult <>0
        then begin
                  clrscr;
                  goto_window(ver);
                  gotoxy(3,4);
                  write('Card Reading Error');
                  write(#7);
```

```
                    Ente._user_password:=false
               end
          else begin
                    read(user_password_file,user_password);
                    close(user_password_file)
               end
end;(*.......... End of  Enter_user_password.......... *)

(* Procedure (4) Enter_sevice_type *)
Procedure Enter_service_type;

(* This Procedure asks for service types .*)
var
   i,j:integer;
begin
     goto_window(hor);
     gotoxy(3,3);
     write('Bring the cursor in the proper row ');
     gotoxy(3,4);
     write('and Press RETURN');
     goto_window(small);
     gotoxy(4,4);
     write('Withdrawl  :');
     gotoxy(4,6);
     write('deposit    :');
     gotoxy(4,8);
     write('Enquiry    :');
     gotoxy(15,4);
     cursor_movement(4,8);
     case wherey of
          4:service_type:=withdrawal;
          6:service_type:=deposit;
          8:service_type:=enquiry
     end
end;(*.......... End of procedure Enter_service_type.............*)


Procedure Service_completion_message(service_type:char);

(* This Procedure gives service completion messages to the user .*)
begin
     goto_window(ver);
     gotoxy(4,2);
     write('Service Over');
     gotoxy(4,4);
     write('Take the slip ');
     gotoxy(4,5);
     write('from the printer');
     gotoxy(4,6);
     write('and keep it .');
     writeln(out,'Transaction Details':60);
     writeln(out,'********************':60);
     writeln(out);
     writeln(out,'Name ':40,user_name:40);
     writeln(out,'Address':40,user_addr:40);
     writeln(out,'Account number':40,ac_no:40);
     writeln(out,'Date of Transaction':40,s:40);
     writeln(out,'Previous Total':40,ptotal:40:2);
     case service_type of
          deposit    :writeln(out,'deposited Amount':40,money:40:2);
          withdrawal:begin
                          n:=trunc(money/type_of_currency);
                          writeln(out,'Withdrawal Amount':40,money:40:2);
                          writeln(out,' ':14,'No of ':7,type_of_currency:3,'Rs':3,'notes issued':13,n:40);
                     end
     end;

     writeln(out,'New Total':40,ntotal:40:2);
     writeln(out);
     flush(out)
end;{.........End of Procedure Service_completion_message.............}
```

```
Procedure Enter_withdrawal_money;

(* This Prrocedure asks for the amount money to be withdrawan from the account
 of the user . *)

begin
     goto_window(hor);
     gotoxy(4,3);
     write('Enter Withdrawal Money');
     gotoxy(4,4);
     write('and Press RETURN');
     goto_window(small);
     gotoxy(3,3);
     write('Withdrawal Money : ');
     read(money);
     goto_window(hor);
     gotoxy(4,3);
     write('Bring the cursor in the correct row and press RETURN.');
     goto_window(small);
     gotoxy(3,2);
     write('Type of Currency');
     gotoxy(5,4);
     write('Rs 5  ');
     gotoxy(5,6);
     write('Rs 10 ');
     gotoxy(5,8);
     write('Rs 100');
    gotoxy(11,4);
     cursor_movement(4,8);
     case wherey of
          4:type_of_currency:=5;
          6:type_of_currency:=10;
          8:type_of_currency:=100
     end;
     clrscr
end;(......Enter_withdrawal_money.......)

(* Procedure (5) withdrawal_processor *)
Procedure withdrawal_processor;
begin
     enter_withdrawal_money;
     ac_file_name[0]:=chr(7);
     ac_file_name:=copy(ac_no,6,3)+'.'+type_of_ac+'ac';
     assign(ac_file,ac_file_name);
     reset(ac_file);
     if ioresult<>0
        then begin
                goto_window(ver);
                gotoxy(3,3);
                write('Error in accessing');
                gotoxy(3,5);
                write('Account file -',ac_file_name);
                goto_window(hor);
                gotoxy(3,3);
                write('Press RETURN');
        end
else begin
                ac_file_size:=filesize(ac_file);
                seek(ac_file,ac_file_size-1);
                read(ac_file,ac_rec);
                ptotal:=ac_rec.total;
                if (ptotal-minamt)<5
                then begin
                                goto_window(ver);
                                gotoxy(3,3);
                                write('your current savings');
                                gotoxy(3,4);
                                write('is Rs ',ptotal:8:2);
                                gotoxy(3,5);
                                write('You cannot withdraw');
                                gotoxy(3,6);
                                write('any money now.');
```

```
                goto_window(hor);
                gotoxy(3,4);
                write('Press RETURN')
          end
else
if (ptotal-money)<minamt
   then begin
                goto_window(ver);
                gotoxy(3,3);
                write('your current savings');
                gotoxy(3,4);
                write('is Rs ',ptotal:8:2);
                gotoxy(3,5);
                write('You cannot withdraw');
                gotoxy(3,6);
                write('more than Rs',ptotal-minamt:8:2,' now');
                goto_window(hor);
                gotoxy(3,4);
                write('Press RETURN')
          end
   else begin
                ntotal:=ptotal-money;
                with ac_rec do
                     begin
                     date:=todate;
                     trans_type:='W';
                     amt:=money;
                     total:=ntotal
                end;
service_completion_message(service_type);
write(ac_file,ac_rec);
seek(bank_file,b);
read(bank_file,bank_rec);
new_sl_no:=bank_rec.sl_no+1;
ntotal:=bank_rec.net_cash+money;
with bank_rec do
     begin
          sl_no:=new_sl_no;
          date:=todate;
          trans_type:='W';
          amt:=money;
          net_cash:=ntotal
      end;
seek(bank_file,b);
write(bank_file,bank_rec);
b:=b+1;
goto_window(hor);
gotoxy(3,3);
write('Press Return ')
                          end

        end;
        close(ac_file)

end;(* .........End of procedure withdrawal_processor.......... *)


Procedure Enter_deposit_money;

(* This Procedure asks for the amount of money to be deposited into the users
  account .*)
begin
     goto_window(hor);
     gotoxy(4,3);
     write('Enter Deposit Money');
     gotoxy(4,4);
     write('and Press RETURN');
     goto_window(small);
     gotoxy(3,4);
     write('Deposit Money  :');
     readln(money);
```

```
end;{......End of Procedure Enter_deposit_money........}


(* Procedure (6) deposit_processor *)

Procedure deposit_processor;
begin
     enter_deposit_money;
     ac_file_name[0]:=chr(7);
     ac_file_name:=copy(ac_no,6,3)+'.'+type_of_ac+'ac';
     assign(ac_file,ac_file_name);
     reset(ac_file);
     if ioresult<>0
        then begin
                  goto_window(ver);
                  gotoxy(3,3);
                  write('Error in accessing');
                  gotoxy(3,5);
                  write('Account file -',ac_file_name);
                  goto_window(hor);
                  gotoxy(3,3);
                  write('Press RETURN')
             end
     else begin
               ac_file_size:=filesize(ac_file);
               seek(ac_file,ac_file_size-1);
               read(ac_file,ac_rec);
               ptotal:=ac_rec.total;
               ntotal:=ptotal+money;
               with ac_rec do
                   begin
                        date:=todate;
                        trans_type:='D';
                        amt:=money;
                        total:=ntotal
                   end;
                write(ac_file,ac_rec);
                service_completion_message(service_type);
                seek(bank_file,b);
                read(bank_file,bank_rec);
                new_sl_no:=bank_rec.sl_no+1;
                ntotal:=bank_rec.net_cash+money;
                with bank_rec do
                     begin
                          sl_no:=new_sl_no;
                          date:=todate;
                          trans_type:='D';
                          amt:=money;
                          net_cash:=ntotal
                     end;
                  seek(bank_file,b);
                  write(bank_file,bank_rec);
                  b:=b+1;
                  goto_window(small);
                  create_window(hor);
                  gotoxy(3,3);
                  write('Press Return ')
             end;
    close(ac_file)
end;{......End of deposit_processor...... }


Procedure Display_enquiries;

(* This Procedure displays different enquiries. *)
begin
     clear;
     create_window(medium);
     create_window(hor);
     gotoxy(3,3);
```

```
      write('Bring the cursor in the proper row');
      gotoxy(3,4);
      write('and Press RETURN');
      goto_window(medium);
      gotoxy(4,2);
      write('(0) Get the details of last transaction.....');
      gotoxy(4,4);
      write('(1) Get the amount of last interest added.....');
      gotoxy(4,6);
      write('(2) Get the no of transaction made this year....');
      gotoxy(4,8);
      write('(3) Get the details of all transactions.....');
      gotoxy(wherex,2);
      cursor_movement(2,8);
      case wherey of
           2:enquiry_type:=0;
           4:enquiry_type:=1;
           6:enquiry_type:=2;
           8:enquiry_type:=3
      end;
      clear;
      create_window(small);
      create_window(hor);
      create_window(ver)
end;{.......End of Display_enquiries........}

Procedure Process_enquiries(i:entype);
var
   j,k,y1,y2:integer;
   done:boolean;

begin
     ac_file_name[0]:=chr(7);
     ac_file_name:=copy(ac_no,6,3)+'.'+type_of_ac+'ac';
     assign(ac_file,ac_file_name);
     reset(ac_file);
     if ioresult<>0
     then begin
          goto_window(ver);
          gotoxy(3,3);
          write('Error in accessing');
          gotoxy(3,5);
          write('Account file ',ac_file_name);
          end
     else begin
          case i of
           0:begin
          ac_file_size:=filesize(ac_file);
          seek(ac_file,ac_file_size-1);
          read(ac_file,ac_rec);
          with ac_rec do
          begin
          writeln(out,'Details of last transaction':60);
          writeln(out,'**************************':60);
          writeln(out);
          writeln(out,'Date of transaction':40,s:40);
          write(out,'Type of transaction':40);
          case trans_type of
          'W':writeln(out,'Withdrawal':40);
          'D':writeln(out,'Deposit':40);
          'I':writeln(out,'Interest Updation':40)
          end;
          writeln(out,'Amount of money involved in transaction':60,amt:20:2);
          writeln(out,'Current Total Savings':40,total:40:2)
          end;


 end;
1:begin
          ac_file_size:=filesize(ac_file);
          j:=ac_file_size-1;
```

```
        done:=false;
        while (not done) or (j=0)
        do begin
                seek(ac_file,j);
                read(ac_file,ac_rec);
                if ac_rec.trans_type='I'
                then begin
                        writeln(out,'Amount of interest last added :Rs',ac_rec.amt);
                        done:=true
                     end
                 else j:=j-1
           end
  end;

2:begin
        ac_file_size:=filesize(ac_file);
        j:=ac_file_size-1;
        k:=0;
        done:=false;
        while (not done) and (j>=0)
        do begin
                seek(ac_file,j);
                read(ac_file,ac_rec);
                val(copy(ac_rec.date,5,2),y1,error);
                val(copy(todate,5,2),y2,error);
                if y1<y2
                then done:=true
                else begin
                        j:=j-1;
                        k:=k+1
                     end
           end;
         goto_window(ver);
         gotoxy(1,3);
         write('No of transactions');
         gotoxy(1,4);
         write('made this year is ',k:4)
  end;
3:begin
       writeln(out,'Date ':15,'Transaction type':30,'Amount':10,'Total':10);
       writeln(out,'-------------------------------------------------------------------------------');
       while not eof(ac_file) do
       begin
            read(ac_file,ac_rec);
            with ac_rec do
            begin
                 writeln(out,change(date):15,map(trans_type):30,amt:10:2,total:10:2);
            end;
       goto_window(ver);
                gotoxy(4,2);
                write('Service Over');
                gotoxy(4,4);
                write('Take the slip ');
                gotoxy(4,5);
                write('from the printer');
                gotoxy(4,6);
                write('and keep it .');
                end
           end;
        end;
     goto_window(hor);
     gotoxy(3,3);
     write('Press RETURN');
     end;
     close(ac_file)
end;(.......End of Procedure Process_enquiries.........)

Procedure Enquiry_processor;
begin
     Display_enquiries;
     process_enquiries(enquiry_type)
end;(*.............. End of procedure Enquiry_processor.............. *)
```

```
(* Function (1) Check_account_no *)
Function check_account_no(s:string8):boolean;

begin
     if upcase(s[1])<> upcase(type_of_ac)
        then begin
                  goto_window(hor);
                  gotoxy(3,4);
                  write('Press RETURN');
                  goto_window(ver);
                  gotoxy(3,3);
                  write('Wrong account number');
                  write(#7);
                  check_account_no:=false
             end
         else begin
                   master_file_name:=copy(s,1,3)+'.DTA';
                   val(copy(s,6,3),act_ac_no,error);
                   assign(master_file,master_file_name);
                   reset(master_file);
                   if ioresult<>0
                      then begin
                                write(#7);
                                goto_window(ver);
                                gotoxy(3,3);
                                write('Master Disk');
                                gotoxy(3,4);
                                write('Reading Error');
                                goto_window(hor);
                                gotoxy(3,4);
                                write('Press RETURN .');
                                check_account_no:=false
                           end
                   else begin
                             seek(master_file,act_ac_no);
                             read(master_file,master_rec);
                             if upcase(master_rec.free) ='N'
                                then begin
                                          check_account_no:=true;
                                          user_name:=master_rec.name;
                                          user_addr:=master_rec.addr;
                                          check_password:=master_rec.pass_field
                                     end
                                else begin
                                          write(#7);
                                          check_account_no:=false;
                                          goto_window(ver);
                                          gotoxy(3,3);
                                          write('Wrong Account no.');
                                          goto_window(hor);
                                          gotoxy(3,4);
                                          write('Press RETURN')
                                     end;
                                end;
                        close(master_file)
                end

end;(*.......... End of Function Check_account_no.............  *)


(* Function user_password_checker Starts *)

Function user_password_checker(i,j:integer):boolean;
begin
     user_password_checker:=i=j

end;
(* End of Function user_password_checker  *)
(*************************************************************************)
                  (*Main user program starts here*)
(*************************************************************************)
```

```
begin
      write(#7);
      clear;
      Display_main;
      wait;
      assign(out,'prn');
      rewrite(out);
      clear;
      create_window(ver);
      create_window(hor);
      gotoxy(3,3);
      write('Enter Date and Press Return.');
      create_window(small);
      gotoxy(4,4);
      write('Date (dd-mm-yy)   :');
      read(s);
      todate[0]:=chr(6);
      todate:=copy(s,1,2)+copy(s,4,2)+copy(s,7,8);
      assign(bank_file,'bank.dta');
      reset(bank_file);
      b:=filesize(bank_file)-1;
      repeat
            clrscr;
            enter_ac_no;
if check_account_no(ac_no)
   then begin
              if enter_user_password then
              if user_password_checker(check_password,user_password)
                 then begin
                           enter_service_type;
                           case service_type of
                                withdrawal:withdrawal_processor;
                                deposit   :deposit_processor;
                                enquiry   :enquiry_processor
                           end;

                      end
                 else  begin
                                goto_window(ver);
                                gotoxy(3,3);
                                write('Wrong Password');
                                goto_window(hor);
                                gotoxy(3,4);
                                write('Press RETURN.');
                       end
        end;
read(kbd,ch);
goto_window(ver);
goto_window(hor);
      until ord(ch)<>13;
      close(bank_file);
      clear;
      close(out)
end.
{*************************************************************************}
                          (Main User Program Ends Here)
{*************************************************************************}
```

T109
30/8/88

```
{$C-}
{$i-}
{$r+}
Program Banker;
{This Program takes care of some of the bank manager's tasks}

{ Constant Declaration for the Main Banker Program }
const
     m_password_file_name                    = 'man.pas';
     user_password_file_name                 = 'a:user.pas';
     status_file_name                        = 'stat.dta';
     emp_file_name                           = 'emp.dta';
     no_file_name                            = 'no.dta';
     bank_file_name                          = 'bank.dta';
     check_m_password                        = 6776;
     no_of_sec                               = 5;
     sec_len                                 = 5;
     max_no_files                            = 99;
     open_account                            = 'O';
     close_ac                                = 'C';
     interest_calculate                      = 'I';
     deposit                                 =  'D';
     savings                                 = 'S';
     current                                 = 'C';
     rate                                    = 0.04;
     small                                   = 1;
     medium                                  = 2;
     big                                     = 3;
     ver                                     = 4;
     hor                                     = 5;
     left_cor                                = 6;
     verybig                                 = 7;
     minamt                                  = 25;


{Type Declaration for the Main Banker Program  }

type
    string3                                  = string[3];
    string4                                  = string[4];
    string5                                  = string[5];
    string6                                  = string[6];
    string7                                  = string[7];
      string8                                                    = string[8];
    string20                                 = string[20];
    string40                                 = string[40];
    master_rec_type                            = record
                                                   date_of_op:string6;
                                                   pass_field:integer;
                                                   free       :char;
                                                   name       :string20;
                                                   addr       :string40
                                                 end;
```

```
ac_rec_type          = record
                         date      :string6;
                         trans_type:char;
                         amt,total :real
                       end;

stat_rec_type        = record
                         file_id ,
                         sec_id    ,
                         no_of_rec :integer
                       end;
bank_rec_type        = record
                         sl_no         :integer;
                         date          :string6;
                         trans_type    :char;
                         amt,net_cash:real
                       end;
emp_rec_type          = record
                         file_id,
                         sec_id,
                         link          :integer
                        end;
emp_f_t              = file of emp_rec_type;
```

```
( Variable Declaration for the Main Banker Program )


var B:integer;
    ch                                                  :char;
    ac_no,s                                             :string8;
    m_password,error,act_ac_no,sec_no,u_password,
    file_no,new_sl_no,no_of_files
                                                        :integer;
    m_password_file,user_password_file                  :file of integer;
    service_request,type_of_ac                          :char;
    master_name                                           :string20;
    master_addr                                           :string40;
    money,ntotal                                        :real;
    todate                                              :string6;
    out                                                 :text;
    master_rec                                          :master_rec_type;
    master_file                                         :file of master_rec_type;
    stat_rec                                            :stat_rec_type;
    stat_file                                           :file of stat_rec_type;
    ac_rec                                              :ac_rec_type;
    ac_file                                             :file of ac_rec_type;
    bank_rec                                            :bank_rec_type;
    bank_file                                           :file of bank_rec_type;
    ac_file_name                                        :string7;
    ac_file_size                                        :integer;
    master_file_name                                    :string8;
    check_password                                      :integer;
    emp_rec                                             :emp_rec_type;
    emp_file                                            :emp_f_t;
    no_file                                             :file of integer;
    over                                                :boolean;

Procedure Create_window(i:integer);
var
   ii,jj,ip,jp:integer;
 Procedure box(left,top,right,bottom:integer);
 var
    x,y:integer;
    procedure twochar(x1,y1,x2,y2:integer);
    begin
         gotoxy(x1,y1);
         write(chr(176));
```

```
            gotoxy(x2,y2);
            write(chr(176))
        end;(twochar)
 begin
      for x:=left to  right do
          twochar(x,top,x,bottom);
      for y:=top to bottom do
        begin
          twochar(left,y,right,y);
          twochar(left+1,y,right-1,y)
        end
 end;(box)
begin
     case i of
          small:begin
                     ii:=5;
                     jj:=2;
                     ip:=51;
                     jp:=15
                end;
          medium:begin
                      ii:=14;
                      jj:=5;
                      ip:=68;
                      jp:=20
                end;
     big:begin
                ii:=4;
                jj:=1;
                ip:=80;
                jp:=24
          end;

 ver:begin
                 ii:=51;
                 jj:=2;
                 ip:=80;
                 jp:=15
             end;
hor:begin

                ii:=5;
                jj:=15;
                ip:=80;
                jp:=24
          end;
 left_cor:begin
                 ii:=1;
                 jj:=1;
                 ip:=30;
                 jp:=5
              end;
          verybig:begin
                      ii:=1;
                      jj:=1;
                      ip:=80;
                      jp:=25
                  end

      end;
     window(1,1,80,25);
     window(ii,jj,ip,jp);
     box(2,2,ip-ii,jp-jj);
     window(ii+4,jj+2,ip-4,jp-2);
     clrscr
end;(......End of Procedure Create_window........)

Procedure Clear;
begin
      window(1,1,80,25);
      clrscr
end;(clear)
```

```
procedure wait;
var
   ch:char;
begin
     read(kbd,ch);
     while ch<>#13 do
           begin
                write(#7);
                read(kbd,ch)
           end
end;(wait)



procedure Display_main;

begin
     clear;
     create_window(verybig);
     gotoxy(2,20);
     write('Press RETURN to start');
     create_window(medium);
     gotoxy(7,3);
     write('AUTOMATED BANK TELLER SYSTEM');
     GOTOXY(10,5);
     WRITE('DESIGNED AND DEVELOPED');
     gotoxy(18,7);
     write('BY');
     gotoxy(12,9);
     write('PURNENDU SINHA');


end;(.....Display_main......)



Procedure cursor_movement(limit_up,limit_down:integer);
var
   ch:char;
   done:boolean;
begin
     done:=false;
     repeat
           read(kbd,ch);
           if ord(ch)=27
              then begin
                        read(kbd,ch);
                        case ch of
                             'P':begin
                                   if wherey=limit_down
                                      then write(#7)
                                      else gotoxy(wherex,wherey+2)
                                 end;
                            'H':begin
                                   if wherey=limit_up
                                      then write(#7)
                                      else gotoxy(wherex,wherey-2)
                                 end
                           else write(#7)
                      end
                 end
           else if ord(ch)=13
                   then done:=true
                   else write(#7)
     until done;
end;(......End of Procedure cursor_movement........)

function power(i:integer):integer;
var
   k,n:integer;
begin
      n:=1;
```

```
      for k:=1 to i do
        n:=n*10;
      power:=n
end;(power)



function convert(i,n:integer):string3;

( This Procedure converts an integer to the corresponding string.  )
var
   k:integer;
   s:string3;
begin
     s[0]:=chr(3);
      for k:=1 to n do
       begin
            s[k]:=chr((i mod power(n)-i mod power(n-1))
                                 div power(n-1)+ord('0'));
            n:=n-1
       end;
     convert:=s;
end;(convert)


Procedure Add(var x:emp_f_t;p:emp_rec_type);
var
   r:emp_rec_type;
begin
     assign(x,'a:emp.dta');
     reset(x);
     seek(x,filesize(x)-1);
     read(x,r);
     r.link:=filesize(x);
     seek(x,filesize(x)-1);
     write(x,r);
     p.link:=maxint;
     write(x,p);
     close(x)
end;(.....Add.....)

Procedure Del(var x:emp_f_t);
var
   r:emp_rec_type;
   i,n:integer;
begin
     assign(x,'a:emp.dta');
reset(x);
seek(x,0);
read(x,r);
if r.link=maxint
   then begin
             close(x);
             assign(no_file,no_file_name);
             reset(no_file);
             read(no_file,n);
             n:=n+1;
             if n>99
                then begin
                          create_window(ver);
                          gotoxy(3,3);
                          write('Account cannot be opened')
                     end
                 else begin
                           seek(no_file,0);
                           write(no_file,n);
                           close(no_file);
                           master_file_name:=type_of_ac+convert(n,2);
                           assign(master_file,master_file_name);
                           rewrite(master_file);
```

```
with master_rec do
     begin
          date_of_op:='';
          pass_field:=0;
          free:='Y';
          name:='';
          addr:=''
      end;
write(master_file,master_rec);
close(master_file);
rewrite(x);
with r do
     begin
          file_id:=n;
          sec_id:=1;
          link:=1
     end;
 for i:=1 to sec_len do
     begin
          write(x,r);
          r.sec_id:=r.sec_id+1;
          r.link:=r.link+1
     end;
  assign(stat_file,status_file_name);
  reset(stat_file);
  seek(stat_file,filesize(stat_file));
  with stat_rec do
       begin
            file_id:=n;
            sec_id:=1;
            no_of_rec:=0
       end;
  for i:=1 to sec_len do
      begin
           write(stat_file,stat_rec);
           stat_rec.sec_id:=stat_rec.sec_id+1
      end
                              end
          end
    else begin
               seek(x,r.link);
               read(x,r);
               seek(x,0);
               write(x,r);
               close(x)
          end
end;(...Del....)
```

```
Function Enter_type_of_account:boolean;

begin
     Enter_type_of_account:=true;
     clrscr;
     create_window(hor);
     gotoxy(3,2);
     write('Bring the cursor in the proper row');
     gotoxy(3,4);
     write('and Press RETURN');
     create_window(small);
     gotoxy(2,2);
     write('Type of Accounts');
     gotoxy(4,4);
     write('Savings   :');
     gotoxy(4,6);
     write('Current   :');
     gotox   5,4);
     curs     vement(4,6);
```

```
      case key of
           5:type_of_ac:=savings;
           6:type_of_ac:=current
           else begin
                     create_window(ver);
                     gotoxy(3,3);
                     write('Improper selection of row');
                     write(#7);
                     Enter_type_of_account:=false
                end

      end;
end;(.....Enter_type_of_account.......)


(* Procedure (2) Enter_user_ac_no *)
Procedure Enter_user_ac_no;
var
   s:string8;
   done:boolean;
   ch:char;
begin
      clrscr;
      create_window(hor);
      gotoxy(3,2);
      write('Enter User Account Number');
      gotoxy(3,4);
      write('and Press RETURN.');
      create_window(small);
      gotoxy(4,4);
      write(' Account No :-');
      read(s);
      ac_no:=s
end;(*......... End of procedure Enter_user_ac_no.......... *)

(* Procedure (3) Enter_user_password *)
function Enter_user_password:boolean;
begin
      Enter_user_password:=true;
      create_window(hor);
      gotoxy(3,2);
      write('Put User Password Card Inside .');
      gotoxy(3,4);
      write('and Press RETURN');
      assign(user_password_file,user_password_file_name);
      reset(user_password_file);
      if ioresult <>0
         then begin
                     clrscr;
                     create_window(ver);
                     gotoxy(4,3);
                     write('Card Reading Error');
                     write(#7);
                     create_window(hor);
                     gotoxy(3,4);
                     write('Press RETURN to try again.');
                     Enter_user_password:=false
               end
          else begin
                     read(user_password_file,u_password);
                     close(user_password_file)
               end

end;(*.......... End of procedure Enter_user_password............ *)

(* Function (1) Check_account_no *)
Function check_account_no(s:string8):boolean;

begin
      if s[1]<>type_of_ac
         then check_account_no:=false
```

```
else begin
          master_file_name[0]:=chr(8);
          master_file_name:=copy(s,1,3)+'.dta';
          val(copy(s,6,3),act_ac_no,error);
          val(copy(s,4,2),sec_no,error);
          assign(master_file,master_file_name);
          reset(master_file);
          if ioresult<>0
          then begin
                    write(#7);
                    create_window(ver);
                    gotoxy(3,3);
                    write('Disk Reading Error');
                    create_window(hor);
                    gotoxy(3,4);
                    write('Press RETURN');
                    check_account_no:=false
               end
          else begin
                    seek(master_file,act_ac_no);
                    read(master_file,master_rec);
                    if upcase(master_rec.free) ='N'
                       then begin
                                 check_account_no:=true;
                                 master_name:=master_rec.name;
                                 master_addr:=master_rec.addr;
                                 check_password:=master_rec.pass_field
                            end
                       else begin
                                 check_account_no:=false;
                                 write(#7);
                                 create_window(ver);
                                 gotoxy(3,3);
                                 write('Wrong Account no');
                                 gotoxy(3,4);
                                 write('Press RETURN')
                            end
                    end;
               close(master_file);
          end
end;
(* End of Function Check_account_no  *)


(* Function Password_checker Starts *)

Function Password_checker(i,j:integer):boolean;
begin
     password_checker:=i=j
end;

function generate_password:integer;
begin
     generate_password:=random(maxint)
end;(.....Generate_password......)

function Enter_manager_password:boolean;
begin
     Enter_manager_password:=true;
     clear;
     create_window(hor);
     gotoxy(3,2);
     write('Put your Password Card Inside');
     gotoxy(3,4);
     write('and Press RETURN');
     wait;
     assign(m_password_file,m_password_file_name);
     reset(m_password_file);
     if ioresult<>0
        then begin
                  clear;
                  create_window(ver);
```

```
                    gotoxy(3,3);
                    write('Card Reading Error');
                    create_window(hor);
                    gotoxy(3,3);
                    write('Press RETURN ');
                    write(#7);
                    wait;
                    Enter_manager_password:=false
               end
          else read(m_password_file,m_password)
end;(.... Enter_manager_password.....)

Procedure Enter_service_request;
begin
     clrscr;
     create_window(hor);
     gotoxy(3,2);
     write('Bring the cursor in the correct row');
     gotoxy(3,4);
     write('and Press RETURN');
     create_window(small);
     gotoxy(4,2);
     write('Specify Type of service needed');
     gotoxy(4,4);
     write('Open Account.....:');
     gotoxy(4,6);
     write('Close Account.....:');
     gotoxy(4,8);
     write('Calculate Interest.....:');
     gotoxy(28,4);
     cursor_movement(4,8);
     case wherey of
          4:service_request:=Open_account;
          6:service_request:=Close_ac;
          8:service_request:=Interest_calculate
     end
end;(......Enter_service_request......)

Procedure Open_account_processor;
var
   i:integer;
   found:boolean;
begin
     if Enter_type_of_account then
     begin
     assign(emp_file,emp_file_name);
     reset(emp_file);
     if ioresult<>0
        then begin
             clear;
             create_window(ver);
             gotoxy(3,3);
             write('Error in accessing');
             gotoxy(3,4);
             write('empty file.');
             write(#7);
             create_window(hor);
             gotoxy(3,3);
             write('Press RETURN to try again');
             over:=false
     end
else begin
             seek(emp_file,0);
             read(emp_file,emp_rec);
             file_no:=emp_rec.file_id;
             sec_no:=emp_rec.sec_id;
             assign(stat_file,status_file_name);
             reset(stat_file);
             if ioresult<>0
                then begin
                          clear;
                          create_window(ver);
```

```
                    gotoxy(3,3);
                    write('Error in accessing');
                    gotoxy(3,5);
                    write('status file.');
                    write(#7);
                    create_window(hor);
                    gotoxy(3,3);
                    write('Press RETURN to try again');
                    over:=false
               end
          else begin
                    i:=(file_no-1)*sec_len+sec_no-1;
                    seek(stat_file,i);
                    read(stat_file,stat_rec);
                    stat_rec.no_of_rec:=stat_rec.no_of_rec+1;
                    seek(stat_file,i);
                    write(stat_file,stat_rec);
                    close(stat_file);
                    if stat_rec.no_of_rec=sec_len
                       then del(emp_file);
                    close(emp_file);
                    master_file_name:=type_of_ac+ copy(convert(file_no,2),1,2)+'.DTA';
                    assign(master_file,master_file_name);
                    reset(master_file);
                    if ioresult<>0 then
begin
     create_window(ver);
     gotoxy(3,3);
     write('Error in accessing');
     gotoxy(3,4);
     write('Master File');
     write(#7);
     over:=false
end

else
begin
i:=sec_len*(sec_no-1);
seek(master_file,i);
found:=false;
while not found do
begin
     read(master_file,master_rec);
     if master_rec.free='Y'
        then found:=true
        else begin
                  i:=i+1;
                  seek(master_file,i)
             end
end;
if found
   then with master_rec do
             begin
                  date_of_op:=todate;
                  pass_field:=generate_password;
                  clrscr;
                  create_window(hor);
                  gotoxy(3,2);
                  write('Put the new card in the drive ');
                  gotoxy(3,4);
                  write('and Press RETURN.');
                  wait;
                  assign(user_password_file,user_password_file_name);
                  rewrite(user_password_file);
                  write(user_password_file,pass_field);
                  close(user_password_file);
                  free:='N';
                  clrscr;
                  gotoxy(3,3);
                  write('Enter user name  ');
                  create_window(small);
                  gotoxy(3,3);
```

```
                        write('Name : ');
                        readln(name);
                        create_window(hor);
                        gotoxy(3,3);
                        write('Enter user address  ');
                        create_window(small);
                        gotoxy(3,3);
                        write('Address : ');
                        readln(addr)
                end;
clrscr;
create_window(hor);
gotoxy(3,2);
write('Take out the card from the drive');
gotoxy(3,4);
write(' and Press RETURN.');
wait;
seek(master_file,i);
write(master_file,master_rec);
close(master_file);
ac_no:=copy(master_file_name,1,3)+
copy(convert(sec_no,2),1,2)+copy(convert(i,3),1,3);
create_window(hor);
gotoxy(3,3);
write('Note the Account Number');
gotoxy(3,4);
write('and Press RETURN');
create_window(ver);
gotoxy(3,3);
write('User Account no');
gotoxy(3,4);
write('is  ',ac_no);
wait;
clrscr;
ac_file_name:=convert(i,3)+'.'+type_of_ac+'ac';
assign(ac_file,ac_file_name);
rewrite(ac_file);
repeat
create_window(hor);
gotoxy(4,4);
write('Enter Deposit Money');
create_window(small);
gotoxy(3,3);
write('Deposit Money : ');
readln(money);
if money<minamt
   then begin
                writeln(out,'Minimum Deposit Money is ',minamt);
                create_window(ver);
                gotoxy(3,3);
                write('Minimum Deposit');
                write('Money is ',minamt)
        end
until money>=minamt;

with ac_rec do
     begin
          date:=todate;
          trans_type:=deposit;
          amt:=money;
          total:=money
     end;
write(ac_file,ac_rec);
close(ac_file);
seek(bank_file,b);
read(bank_file,bank_rec);
new_sl_no:=bank_rec.sl_no+1;
ntotal:=bank_rec.net_cash+money;
with bank_rec do
     begin
          sl_no:=new_sl_no;
          date:=todate;
```

```
                                    trans_type:='D';
                                    amt:=money;
                                    net_cash:=ntotal
                                end;
                            write(bank_file,bank_rec);
                            b:=b+1;
                            clrscr;
                            create_window(ver);
                            gotoxy(3,3);
                            write('Service Over');
                            writeln(out,'User Details':50);
                            writeln(out,'------------':50);
                            writeln(out);
                            writeln(out,'Name':40,' :',master_rec.name:40);
                            writeln(out,'Address':40,' :',master_rec.addr:40);
                            writeln(out,'Account No':40,' :',ac_no:40);
                            writeln(out,'Date of Opening':40,' :',s:40);
                            writeln(out,'Amount Deposited':40,' :',money:40:2)

                        end;
                    end
        end;
    end
   else over:=false
end;{.......Open_Account_Processor.........}

Procedure Calculate_interest;
const
     max_no_of_trans                          = 100;
type
    auxt =record
                dfield :string6;
                tot    :real
          end;
var
   no_trans,i,k,n,d :integer;
   auxarr      :array[0..max_no_of_trans] of auxt;
   auxrec      :auxt;
   complete    :boolean;
   locmin,interest:real;
   month,premonth:string[2];
begin
     interest:=0;
      k:=filesize(ac_file)-1;
      complete:=false;
      no_trans:=0;
      while (not complete) and (k>=0)
      do begin
              seek(ac_file,k);
              read(ac_file,ac_rec);
          if ac_rec.trans_type='I'
             then begin
                       complete:=true;
                       no_trans:=no_trans+1
                  end
             else begin
                       auxarr[no_trans].dfield:=ac_rec.date;
                       auxarr[no_trans].tot:=ac_rec.total;
                       k:=k-1;
                       no_trans:=no_trans+1
                  end
    end;
locmin:=auxarr[no_trans-1].tot;
month:=copy(auxarr[no_trans-1].dfield,3,2);
for n:=no_trans-2 downto 0
    do begin
              val(copy(auxarr[n].dfield,1,2),d,error);
              month:=copy(auxarr[n].dfield,3,2);
              if premonth<>month
                 then begin
                           interest:=interest+locmin*rate/12;
                           locmin:=auxarr[no_trans].tot
```

```
                        end
                    else if d>10
                        then if auxarr[n].tot<locmin
                            then locmin:=auxarr[n].tot
            end;
        seek(ac_file,filesize(ac_file)-1);
        read(ac_file,ac_rec);
        ac_rec.total:=ac_rec.total+interest;
        ac_rec.amt:=interest;
        ac_rec.date:=todate;
        ac_rec.trans_type:='I';
        seek(ac_file,filesize(ac_file)-1);
        write(ac_file,ac_rec)
end;{.........Calculate_Interest.............}

Procedure Close_account(s3:string3;i,j:integer);
var
   k,file_no:integer;
begin
        ac_file_name[0]:=chr(7);
        ac_file_name:=copy(ac_no,6,3)+'.'+type_of_ac+'ac';
        assign(ac_file,ac_file_name);
        reset(ac_file);
        if ioresult<>0
        then begin
                clear;
                create_window(ver);
                gotoxy(3,3);
                write('Error In Reading');
                gotoxy(3,4);
                write('Account File');
                write(#7);
                create_window(hor);
                gotoxy(3,3);
                write('Press RETURN to try again');
                over:=false
        end
else begin
                calculate_interest;
                create_window(ver);
                gotoxy(3,4);
                write('Service Over');
                writeln(out);
                writeln(out,'Date of Closing':40,s:40);
                writeln(out,'Amount of Money To Be Returned':40,ac_rec.total:40:2);
                close(ac_file);
                erase(ac_file);
                assign(master_file,master_file_name);
                reset(master_file);
        if ioresult<>0
           then begin
                create_window(ver);
                gotoxy(3,3);
                write('Error In Reading');
                gotoxy(3,4);
                write('The Master file');
                close(master_file);
             end
        else begin
                seek(master_file,act_ac_no);
                master_rec.free:='Y';
                write(master_file,master_rec);
                close(master_file);
                assign(stat_file,status_file_name);
                reset(stat_file);
                if ioresult<>0
                   then begin
                        create_window(ver);
                        gotoxy(3,3);
                        write('Error In Reading Status File');
                        close(status_file);
                     end
```

```
                                        else begin
                                                val(copy(s3,2,2),file_no,error);
                                                k:=(file_no-1)*sec_len+sec_no-1;
                                                seek(stat_file,k);
                                                read(stat_file,stat_rec);
                                                with stat_rec
                                                 do no_of_rec:=no_of_rec-1;
                                                seek(stat_file,k);
                                                write(stat_file,stat_rec);
                                                close(stat_file)
                                           end
                                end
                        end
end;(.....Close_account......)

Procedure Close_account_processor;
begin
      if Enter_type_of_account then
      begin
       Enter_user_ac_no;
       if check_account_no(ac_no)
          then begin
                    if Enter_user_password
                       then if password_checker(check_password,u_password)
                               then close_account(master_file_name,sec_no,act_ac_no)
                               else begin
                                       clear;
                                       create_window(ver);
                                       gotoxy(3,3);
                                       write('Wrong Password');
                                       create_window(hor);
                                       gotoxy(3,3);
                                       write('Press RETURN to try again.');
                                       write(#7);
                                       over:=false
                                    end
               end
          else begin
                    clear;
                    create_window(ver);
                    gotoxy(3,3);
                    write('Wrong Account no');
                    write(#7);
                    over:=false
               end
      end
      else over:=false

end;(......Close_account_processor........)

Procedure Interest_calculation_processor;
var
   i:integer;
begin
     if Enter_type_of_account then
        begin
               Enter_user_ac_no;
               if check_account_no(ac_no) then
               begin
               ac_file_name[0]:=chr(7);
               ac_file_name:=copy(ac_no,6,3)+'.'+type_of_ac+'ac';
               assign(ac_file,ac_file_name);
               reset(ac_file);
               if ioresult<>0
                  then begin
                             create_window(ver);
                             gotoxy(3,3);
                             write('Error in accessing');
                             gotoxy(3,4);
                             write('Account file.');
                             create_window(hor);
                             gotoxy(3,4);
```

```
                        write('Press RETURN');
                        over:=false
                    end
                else begin
                        calculate_interest;
                        write(ac_file,ac_rec);
                        create_window(small);
                        create_window(ver);
                        gotoxy(4,5);
                        write('Service Over');
                    end;
                close(ac_file)
            end
        end
    else over:=false

end;{.....Interest_calculation_processor.....}

Function Check_manager_password(i:integer):boolean;
begin
     check_manager_password:=check_m_password=i

end;{.....Check_manager_password.....}

{*************************************************************************}

begin
     clear;
     Display_main;
     wait;
     assign(out,'prn');
     rewrite(out);
     clear;
     create_window(hor);
     gotoxy(3,3);
     write('Enter Date and Press RETURN.');
     create_window(small);
     gotoxy(4,4);
     write('Date (dd-mm-yy) :');
     read(s);
     todate[0]:=chr(6);
     todate:=copy(s,1,2)+copy(s,4,2)+copy(s,7,8);
     if enter_manager_password
        then if check_manager_password(m_password)
             then begin
                     assign(bank_file,bank_file_name);
                     reset(bank_file);
                     b:=filesize(bank_file)-1;
                     seek(bank_file,b);
                  repeat
                  clear;
                  over:=true;
                  enter_service_request;
                  case service_request of
                       open_account        :open_account_processor;
                       close_ac            :close_account_processor;
                       interest_calculate :interest_calculation_processor
                  end;
                  if over
                     then begin
                             create_window(hor);
                             gotoxy(3,3);
                             write('Press RETURN for further use')
                          end;
                  read(kbd,ch)
              until ch<>#13
             end
     else begin
               create_window(ver);
               gotoxy(3,3);
               write('Wrong Password ');
               write(#7);
```

```
                        create_window(hor);
                        gotoxy(3,3);
                        write('Press RETURN');
                        wait
                    end;
        close(bank_file);
        clear
end.

(*************************************************************************)
h*                    Main Banker Program Ends Here                      *)
(*************************************************************************)
```