Computer Program: General University Requirements Package

Abhishek Kejriwal '93

Illinois Wesleyan University
The objective of my research project was to write a computer program in Turbo Pascal which would determine how many general university requirements a student has completed and what requirements he or she needs to complete in order to graduate. There are six degrees offered at Illinois Wesleyan University. They are: BA (Bachelor of Arts); BS (Bachelor of Science); BFA (Bachelor of Fine Arts); BSN (Bachelor of Science in Nursing); BM (Bachelor of Music) and BME (Bachelor of Music Education). There is a different set of criteria to be met for the completion of each of these degrees. The program processes student records and generates the appropriate check form.

Coding this problem and generating the output were extremely difficult because there are several classes, sub-classes, permutations and combinations possible to satisfy a requirement. Just to give a flavor of the complexity I will give an example. As stated earlier there are six degrees, each with different requirements. One of them is the BA. Humanities is one of thirteen requirements a student has to meet to complete the BA degree. To meet the Humanities requirement the student must complete three courses from at least two of the following areas: Literature, Philosophy and Humanities. There are seven successful ways to meet this requirement. A couple of these are two courses in Literature and one in Philosophy or two in Literature and one in Humanities and so on. Further, there are about 29 courses in Literature, 23 courses in Philosophy and 5 courses in Humanities that qualify. In addition to this, the program has to check whether the course is valid. For a course to be valid, the course grade should not be Credit, No Credit, Withdrawn, Pass, Fail, Incomplete or Dropped and it should have a unit value of 0.7 or
more. If the parts in the problem were mapped in a tree format there would be an incredible number of branches in the end. Ultimately there was the question of testing. To be sure that a program is working correctly one must perform a number of test runs. Some computer scientists describe testing as the most important part of the program. It was necessary to type in the records of students and generate results and then match the output to the results computed manually. Several such records had to be entered and any errors generated had to be ironed out. After a considerable amount of test data the package was finally generating outputs which exactly matched the results of outputs generated manually.

This program will be used in the Registrar's office at Illinois Wesleyan University starting this summer. After each semester the staff at the registrar's office will simply update the already existing data-base by adding any new students or adding courses to the records of the existing students. Copies of the form generated by the program after processing the checks will be sent to each student's advisor. Previously this entire process was accomplished manually and was extremely time consuming. With the help of this program the advisors will know at a glance where their advise's stand in terms of completing graduation requirements.
Instructions

on how to use the

General University

Requirements Package

Developed by

Abhishek Kejriwal
INSTRUCTIONS

I) Start your computer.

II) At C prompt type in 'CHECK1' if you want to perform checks for the BA or BS degree, and then press enter. Type in 'CHECK2' if you want to perform checks for the BFA, BM, BME or BFA degrees, and then press enter.

III) When you see the menu and 'Enter your Selection' flashing press the Caps Lock button. Leave the button on throughout the use of the program.

IV) Enter your selection:

1. For adding a student record.
2. For deleting a student record.
3. For editing an existing student record.
4. For viewing a present student record on the screen.
5. For printing a student record.
Q. To quit.

1. Adding a Student Record

a) Please enter the students Social Security Number. (Remember no dashes; only the nine digits)

Error Conditions: (i) Social Security has to be exactly nine digits long.
(ii) Social Security number entered already exists.

b) Please enter the Name, Advisor's Name, Major, School and Degree at the appropriate prompts. For a double major, separate the two majors by a '/'. The order in which they are entered is not important.

Caution: If you make an error do not panic, you will be given a chance to correct any errors at the end of all five inputs.

If your inputs are okay then answer 'Y' to the question are all inputs okay. If not, answer 'N'. If you answer 'Y' then you will be asked if you want to enter courses. If you answer no you will be taken to the edit menu. Choose and edit the appropriate data field. After that you will be given a chance to enter courses. If you want to enter courses answer 'Y' when prompted, else answer 'N'.

If you answer 'N' you will go back to the main menu.

If you answer 'Y' you will have to enter information about a course.

(i) Course Dept. - Enter the appropriate name as given in the registration booklet.

(ii) Course Number - Number should be greater than 99 and less than 500. Be very careful!!!! If you make an error here you will have to start all over again.

(iii) Course Value - You can enter any value from 0.0 to 1.90. Entering the number 1 without any decimal point is acceptable. You can also enter 'X' or 'Y' for the P.E. courses.

(iv) Course Grade - You can enter A,B,C,D,F,P,IN,CR,NCR and W as appropriate.

Caution: If you make an error do not panic, you will be given a chance to correct any errors at the end of all four inputs.

Enter as many courses as required. After you are done you will come back to the main menu.

2. Deleting a Student Record

a) Please enter the students Social Security Number. (Remember no dashes; only the nine digits)

Error Conditions : (i) Social Security has to be exactly nine digits long.

(ii) Social Security number entered does not exist, check number and enter again.

3. Editing an existing record

a) Please enter the students Social Security Number. (Remember no dashes; only the nine digits)

Error Conditions : (i) Social Security has to be exactly nine digits long.

(ii) Social Security number entered does not exist, check number and enter again.
(b) First, you will be given a menu where you can alter the Name of the student, Advisor's Name, Major, School or Degree. Make changes as required and then choose done to continue. If you do not need to make any changes select done.

(c) Now, you will be given a chance to add or delete courses from the existing list of courses. To add courses choose '1' and then follow the add procedure.

To delete courses you need to select '2'. To delete a course you only need to specify the course department and course number.

Error Conditions: (i) Course does not exist. Check list and try again.

(d) To get back to the main menu, select '3'.

4. Viewing an existing record on the screen

a) Please enter the students Social Security Number. (Remember no dashes; only the nine digits)

Error Conditions: (i) Social Security has to be exactly nine digits long.

(ii) Social Security number entered does not exist, check number and enter again.

Caution: In order to freeze a screen, Press the Pause button. To continue press the enter key. You do not need to press the enter key for any other reason. The screens will automatically advance after a brief pause.

5. Printing an existing record

a) Please enter the students Social Security Number. (Remember no dashes; only the nine digits) Make sure the printer is on line and the paper is positioned properly.

Error Conditions: (i) Social Security has to be exactly nine digits long.

(ii) Social Security number entered does not exist, check number and enter again.

(iii) Printer is not ready.
GENERAL UNIVERSITY REQUIREMENTS

O.K.  I.  Writing:  English 105 or equivalent: ENGL 105

II.  Fine Arts: _____

O.K.  III.  Humanities:

A.  Logic: PHIL 102
B.  Religion: REL 225
C.  Philosophy: One course in philosophy chosen from 103, 104, 109, 110, 111, 120, 270, 271, or an appropriate 112: PHIL 270
D.  Humanities, Literature: one course: ENGL 270

O.K.  IV.  Social Science:

A.  Social Science 102: SOSC 102
B.  Two course units chosen from:
   Economics:
   History: HIST 204
   Political Science: PSCI 327
   Sociology (not 291): _____

O.K.  V.  Natural Science:

A.  Human Biology 107 and 108: BIOL 107 BIOL 108
B.  Chemistry 110: CHEM 110
C.  Natural Science: NASC 101

VI.  Mathematics or Computer Science: _____

VII.  Physical Education:

2 courses (X) or 4 half courses (Y) or an equivalent: combination: _____ _____ _____ _____

3.00 Total completed (11 course units numbered 300 and 400 are required with at least 4 in a departmental major)

23.00 Total units completed (35 course units required)

6.00 Total "D" units (no more than 4 will be counted toward graduation)

1.00 Total "D" units in major1: NURS (no more than 1 will be counted toward graduation)

Student Signature: _____________________ Date: ______________

Advisor Signature: _____________________ Date: ______________
GENERAL UNIVERSITY REQUIREMENTS

I. Writing: English 105 or equivalent: _____

II. Foreign Language: _____

III. Fine Arts: _____

IV. Humanities:

Three courses from at least two of the following areas:
1. Literature ______
2. Philosophy PHIL 111 ______
3. Humanities HUM 111 ______

O.K.

V. Natural Science:

A. One life science: BIOL 108
   (Biology, Psychology, NatSci 101)
B. One Physical Science: CHEM 101
   (Chemistry, Physics)
C. One Laboratory Science: CHEM 101

O.K.

VI. Social Science:

Three courses from three different areas:

   Economics: ______
   History: ______
   Political Science: PSCI 102
   Sociology (not 291): ______
   Social Science: ______

O.K.

VII. Mathematics or Computer Science: CS 111

VIII. Physical Education:

   2 courses (X) or 4 half courses (Y) or an equivalent:
   combination: ______ ______ ______ ______

XI. Religion: ______

4.00 Total completed (11 course units numbered 300 and 400 are required with at least 4 in a departmental major)

19.00 Total units completed (35 course units required)

0.00 Total "D" units (no more than 4 will be counted toward graduation)

0.00 Total "D" units in major1: MATH (no more than 1 will be counted toward graduation)

0.00 Total "D" units in major2: CS (no more than 1 will be counted toward graduation)
Major Requirements

Major Requirements Completed

Major1 : MATH

1. MATH 389
2. MATH 161
3. MATH 421

Major2 : CS

1. CS 111
2. CS 112
3. CS 350
4. CS 380

Major Requirements to be completed

Student Signature: ________________  Date: ___________
Advisor Signature: ________________  Date: ___________
ILLINOIS WESLEYAN UNIVERSITY
BFA CREDIT CHECK FORM

Student Name: JAMES, JOHN
Major 1: MOTH
Advisor Name: DR. LOITZ
Major 2: NONE

GENERAL UNIVERSITY REQUIREMENTS

O.K. I. Writing: English 105 or equivalent: ENGL 105

_____ II. Foreign Language: _____

_____ III. Religion: _____

O.K. IV. Humanities:

Two courses from at least two of the following areas:

1. Literature ENGL 363
2. Philosophy PHIL 112
3. Humanities

_____ V. Social Science:

Two courses from two different areas:

Economics: _____
History: _____
Political Science: _____
Sociology (not 291): SOC 120
Social Science: _____

_____ VI. Natural Science: at least two course units from at least two of the following groups

Group 1: (biology, psychology, NatSci 101) BIOL 175
Group 2: (chemistry, physics) _____
Group 3: (mathematics, computer science) _____

O.K. VII. Physical Education:

2 courses (X) or 4 half courses (Y) or an equivalent:
combination: PEC 141 PEC 142 _____

9.00 Total completed (11 course units numbered 300 and 400 are required with at least 4 in a departmental major)
32.25 Total units completed (34 course units required)
2.00 Total "D" units (no more than 4 will be counted toward graduation)
0.00 Total "D" units in major 1: MOTH (no more than 1 will be counted
0.00 Total "D" units in major 2: NONE (no more than 1 will be counted toward graduation)

Student Signature: ____________________ Date: __________
Advisor Signature: ____________________ Date: __________
GENERAL UNIVERSITY REQUIREMENTS

O.K.  I. Writing: English 105 or equivalent: ENGL 105

O.K. II. Foreign Language: SPAN 201 SPAN 201

O.K. III. Religion: REL 202

O.K. IV. Humanities:

Two courses from at least two of the following areas:

1. Literature ENGL 270
2. Philosophy
3. Humanities

O.K. V. Social Science:

Two courses from two different areas:

Economics: 
History: HIST 201
Political Science: 
Sociology (not 291): SOC 120
Social Science: 

O.K. VI. Natural Science: at least 3 course units from (biology chemistry, or physics) and at least one laboratory science:

CHEM 130 BIOL 117 CHEM 130 PHYS 239

O.K. VII. Physical Education:

2 courses (X) or 4 half courses (Y) or an equivalent:
combination: PEC 122 PEC 129

O.K. VIII. Mathematics: one course unit MATH 100

6.00 Total completed (11 course units numbered 300 and 400 are required with at least 4 in a departmental major)

36.70 Total units completed ( 39 course units required)

0.00 Total "D" units (no more than 4 will be counted toward graduation)

0.00 Total "D" units in major1 : MUSED (no more than 1 will be counted toward graduation)

0.00 Total "D" units in major2 : NONE (no more than 1 will be counted toward graduation)

Student Signature: __________________ Date: _____________

Advisor Signature: __________________ Date: _____________
ILLINOIS WESLEYAN UNIVERSITY
BM CREDIT CHECK FORM

Student Name: BARD, PATRICK
Major1: MUS
Advisor Name: DR. DAVID NOTT
Major2: NONE

GENERAL UNIVERSITY REQUIREMENTS

O.K.  I. Writing: English 105 or equivalent: ENGL 105

_____ II. Foreign Language: ______ ______

O.K. III. Religion: REL 101

O.K. IV. Humanities:

Two courses from at least two of the following areas:

1. Literature ______
2. Philosophy ______
3. Humanities HUM 375

O.K. V. Social Science:

Two courses from two different areas:

Economics: ______
History: HIST 205
Political Science: ______
Sociology (not 291): SOC 120
Social Science: ______

_____ VI. Natural Science: at least two course units from at least two of the following groups

Group 1: (biology, psychology, NatSci 101) PSYC 100
Group 2: (chemistry, physics) ______
Group 3: (mathematics, computer science) ______

_____ VII. Physical Education:

2 courses (X) or 4 half courses (Y) or an equivalent:
combination: PEC 124 ______ ______

O.K. VIII. Elective: 1 course unit non-music elective GER 101

1.00 Total completed (11 course units numbered 300 and 400 are required with at least 4 in a departmental major)

9.00 Total units completed (37 course units required)

1.00 Total "D" units (no more than 6 will be counted toward graduation)

0.00 Total "D" units in major1: MUS (no more than 1 will be counted toward graduation)

0.00 Total "D" units in major2: NONE (no more than 1 will be counted toward graduation)

Student Signature: ____________ Date: ____________

Advisor Signature: ____________ Date: ____________
Program Project (Input,Output);

Uses Crt,Dos,Printer,Draw,Copy;

Type Darray = array[1..2000] of string[11]; { Stores the SS# 's of all students}
(The following record structure stores information about a particular course)

Type CourseInfo = Record
    Dept:string[20];
    Number: integer;
    Value:string[4];
    Grade:string[3];
End;

{This Record structure is required to store results of any check}

Type Proces = Record
    Depts:string;
    Numbers: integer;
End;

{ Array of course information }

Type CourseArray = Array [1..80] of Courseinfo; {stores all the students courses}

Var Course:CourseInfo; {course information}

{The following are arrays of course information}

{Coursearray holds all the courses of a student}
{Mcoursearray holds all the major1 courses and Mcoursearray2, major2 courses}

Var CourseArray, Mcoursearray, Mcoursearray2 :CourseArray;

Var Adder:Real; {Holds the value when strings are converted to reals}
Var Code:integer; {Checks if the conversion was error free}

{The next three lines of variables contain the results of checks}

Var Eng, MaCs, Life, Lit1, Lit2, Phil1, Hum1, PhSc, LabSc, Rel : Proces;
Var Econ, Hist, PSCI, Soc, soSc, Pec1, Pec2, Pec3, Pec4, Phil2, Hum2 : Proces;
Var Fart, Flang, Flang2, Flang3, Math, Elec, NS1, NS2, NS3 : Proces;
Var Log, SS102, PhilN, HumLit, Bio107, Bio108, Chem110, NS101 : Proces;

{ I, J, K, Counter, Count and Counting are counters}

Var I,J,K,Counter, Count, Counting:integer;

Var Tunit, Dmajor, Dmajor2, Higher, Dunit :real;

Var Datafile:text; {Contains the SS# of all the students}
Var Infofile:text; {Contains information about the student being processed}
Var Datarray:darray; {Contains the SS# of all students}
Var Stringy, Temp, Num:string[12]; {contain SS# during transitions}
Var Name, Advisor:string[30]; {Contains the name of student as so on}
Var Major, Major2:string[25];

Var Degree:string[5];

Var School:string[15];
Var Entrance:string[15];

Var Mcheck, Check, Request1, Request2, Find:Boolean; {Help in checking whether}
{certain checks need to be performed}

Var Choice, Sure, New:Char; {Sure is used make sure if you want to delete}
Var NScount, NScount1, SScount, PeCount, Hcount : integer; {Keep track of how many Natural Science and Humanities requirements have been completed}

Var Eight : string[8]; {Helps in converting the SS#}

{This procedure helps preventing the user from overwriting a record}
{It checks if the record already exists}
Procedure Exists;

Begin
  new:=‘y’;
  count:=1; { Count is used as a counter }
  find:=false; { Counter is the number of records present }
  While ((datarray[counter]<>datarray[Count]) and (Count<(Counter-1))) do
    Begin
      count:=count+1;
    End;
  If datarray[counter]=datarray[count] then
    Begin
      find:=true;
      datarray[counter]:={
    End;
  If find=true then
    Begin
      writeln; writeln;
      new:={
      sound(700) ;
      delay(100);
      nosound;
      writeln(‘Record already exists !!!!!!!!!!!!!!!!’:58);
      writeln;
      writeln;
      write(‘Do you want to overwrite (Y/N) : ’);
      readln(New);
    End;
End;

{ This procedure prints out the menu }
PROCEDURE PrintMenu(Var Choice:Char);
{Print the menu of use options, and determine the user’s choice}
Begin
  Clrscr;
  sound(500);
  delay(100);
  nosound;
  Drawbox(12,66,1,23,4);
  Gotoxy(15,2); Textcolor(9);
  Write (‘R E C O R D   S E L E C T I O N’); Textcolor(7);
  Gotoxy(15,5); Write(‘1. Add a student record’); TextColor(9);
  Gotoxy(15,7); Write(‘2. Delete a student record’);Textcolor(7);
  Gotoxy(15,9); Write(‘3. Edit a student record’);Textcolor(3);
  Gotoxy(15,11); Write(‘4. View a student record’);Textcolor(5);
  Gotoxy(15,13); Write(‘5. Print a student record’);
  Textcolor(7);
  Gotoxy(15,15); Write(‘Q. Quit’);Textcolor(15+126);
  Gotoxy(15,19); Write(‘ENTER YOUR SELECTION: ’);
  Readln(Choice);
  Textcolor(13);
  writeln(chr(7));

if choice in ['1','2','3','4','5','Q','q'] then write('')
else
    begin
clear;
    choice:='6';
end;
end;(Printmenu)

{***************************************************************}
{ This procedure initializes all the variables }
{ For explanation of variables look at declaration section }

Procedure Initializei
Begin
    Name:='';
    Advisor:='';
    Major:='';
    Major2:='NONE';
    Degree:='';
    School:='';
    Entrance:='';
    for I:=1 to 80 do
        Begin
            CourseArray[I].Dept:='';
            CourseArray[I].Number:=0;
            CourseArray[I].Grade:='';
            CourseArray[I].Value:='';
        End;
    for I:=1 to 14 do
        Begin
            Mcoursearray[I].Dept:='';
            Mcoursearray2[I].Dept:='';
            End;
    MaCs.Depts:=''; MaCs.Numbers:=0;
    Rel.Depts:=''; Rel.Numbers:=0;
    Hist.Depts:=''; Hist.Numbers:=0;
    Econ.Depts:=''; Econ.Numbers:=0;
    PSCI.Depts:=''; PSCl.Numbers:=0;
    Soc.Depts:=''; Soc.Numbers:=0;
    sosc.Depts:=''; sosc.Numbers:=0;
    Flang.Depts:=''; Flang.Numbers:=0;
    Flang2.Depts:=''; Flang2.Numbers:=0;
    Life.Depts:=''; Life.Numbers:=0;
    PhSc.Depts:=''; PhSc.Numbers:=0;
    LabSc.Depts:=''; LabSc.Numbers:=0;
    Pec1.Depts:=''; Pec1.Numbers:=0;
    Pec2.Depts:=''; Pec2.Numbers:=0;
    Pec3.Depts:=''; Pec3.Numbers:=0;
    Lit1.Depts:=''; Lit1.Numbers:=0;
    Lit2.Depts:=''; Lit2.Numbers:=0;
    Phil1.Depts:=''; Phil1.Numbers:=0;
    Phil2.Depts:=''; Phil2.Numbers:=0;
    Hum1.Depts:=''; Hum1.Numbers:=0;
    Hum2.Depts:=''; Hum2.Numbers:=0;
    Fart.Depts:=''; Fart.Numbers:=0;
    Elec.Depts:=''; Elec.Numbers:=0;
    NS1.Depts:=''; NS1.Numbers:=0;
NS2.Depts:=''; NS2.Numbers:=0;
NS3.Depts:=''; NS3.Numbers:=0;
Log.Depts:=''; Log.Numbers:=0;
SS102.Depts:=''; SS102.Numbers:=0;
PhilN.Depts:=''; PhilN.Numbers:=0;
HumLit.Depts:=''; HumLit.Numbers:=0;
NS101.Depts:=''; NS101.Numbers:=0;
End;
{*************************************************************************}
{ This procedure reads all the data from Datafile }
{ Datafile is the one that has all the Social Security numbers are 'Counter'}
{*************************************************************************}
{ Procedure Readata }
Begin
for I:= 1 to 2000 do { Initializes the array containing the SS# }
  Datarray[I]:='';
  Stringy:='';
  Counter:=1;
  while (stringy <> '') do 
    Begin
      readln(Datafile,Stringy);
      if stringy=' ' then write('')
      else
        Begin
          Datarray[Counter]:=Stringy;
          Counter:=Counter+1;
        End;
    End;
  Counter:=Counter-2;
End;
{*************************************************************************}
{ This procedure is called if there is an error in inputting data about a particular student or course. It takes care of all the editing }
Procedure EditRecord;
Var Select, Corse, Choose:Char; {Flag variables for loops}
Var Selection, Checking:Boolean; {Flag variables for loops}
Begin
  selection:=false;
  select:=' ';
  While Select<>’6’ do 
    Begin
      selection:=false;
      (The following loop helps edit the data about a particular student) 
      (For example if the name of the student is inputted wrong)
      While ((not(selection)) and (Request1)) do
        Begin
          clsclr;
          writeln; writeln;
          writeln('Input what field you want to edit ':54);
          writeln('--------------------------------- ':54);
          writeln; writeln;
        End;
    End;
End;
write(''1'' : Name of Student : ':42); writeln(Name);
write(''2'' : Name of Advisor : ':42); writeln(Advisor);
write(''3'' : Major : ':42); writeln(Major);
write(''4'' : School : ':42); writeln(School);
write(''5'' : Degree : ':42); writeln(Degree);
write(''6'' : Done : ':42);
writeln; writeln;
write('Input Choice ':42);
readln(Select); {Helps select which field needs to be edited}
write(chr(7));
Case select of
'1': Begin
  writeln;
  write('Please input the edited name of the student : ');
  readln(Name);
  write(chr(7));
  selection:=true;
End;
'2': Begin
  writeln;
  write('Please input the edited name of the advisor : ');
  readln(Advisor);
  write(chr(7));
  selection:=true;
End;
'3': Begin
  writeln;
  write('Please input the new major : ');
  readln(Major);
  write(chr(7));
  selection:=true;
End;
'4': Begin
  writeln;
  write('Please input the new school : ');
  readln(School);
  selection:=true;
  write(chr(7));
End;
'5': Begin
  writeln;
  write('Please input the new degree : ');
  readln(Degree);
  write(chr(7));
  selection:=true;
End;
End; {End of Case}
if select='6' then
begin
  write('');
  selection:=true;
end;
If selection=false then
Begin
  clrscr;
  writeln; writeln;
  write('Choice not found !!!!!!!!!!!!!');
  write(chr(7));
  delay(3000);
End;  {End of While}
End;  {End of Select}
corse:=' ';  
If request2=true then

Begin

{The following loops helps edit information about a particular course}
{You can also add or delete from the present list of courses }

While corse<>'3' do
Begin
clrscr;
writeln('Present List of Courses ':52);
writeln('----------------------- ':52);
writeln;
writeln(' Department Number Unit Grade');
writeln;
for I:= 1 to counting do 
Begin
If coursearray[I].Grade<>'
then
Begin
write(coursearray[I].dept:14);  
write(coursearray[I].Number:17);  
write(coursearray[I].value:16,coursearray[I].grade:15);  
writeln;
If I mod 13 = 0 then
Begin
writeln; writeln;
writeln('Press Return to Continue...':78); 
while not (keypressed) do;
readln;
writeln; writeln;
clrscr;
writeln('Present List of Courses ':52);
writeln('----------------------- ':52);
writeln;
writeln(' Department Number Unit Grade');
writeln;
End;
End;
writeln; writeln;
writeln('Input "1" if you want to add courses to the list : ');
writeln('Input "2" if you want to delete courses to the list : ');
writeln('Input "3" if you want to make no changes : ');
writeln; writeln;
write('Input Choice : ');
readln(Corse);
write(chr(7));
choose:='y';
write(chr(7));
writeln; writeln;
Case Corse of
   
   '1': Begin
   While choose in ['y','Y'] do
begin
clrscr;
writeln; writeln; writeln;
counting:=counting+1;
writeln('Input Department : ':44);
writeln('(Note: All Uppercase) ':44);
GotoXY(45,4);
readln(CourseArray[Counting].Dept);
writeln; writeln;
write('Input Course Number : ':44);
readln(CourseArray[Counting].Number);
writeln;
write(' Input Course Unit : ':44);
readln(CourseArray[Counting].Value);
writeln;
write(' Input Course Grade : ':44);
readln(CourseArray[Counting].Grade);
writeln;
write('Do you want to enter more courses (Y/N) : ');
readln(Choose); write(chr(7));
end;

{ Input '2' if you want to delete from the present list of courses }

'2': Begin
While choose in ['y','Y'] do
begin
clrscr;
checking:=false;
writeln; writeln; writeln;
write('Input Department , : 44} ;
readln(Course.dept);
writeln;
write('Input Course Number , : 44} ;
readln(Course.Number);
writeln;
writeln;
for i:= 1 to counting do
if ((coursearray[I].dept=course.dept) and (coursearray[I].Number= course.number))
then
Begin
   coursearray[I].dept:=' '
   coursearray[I].number:=0;
   coursearray[I].grade:=' '
   coursearray[I].Value:=' '; 
   checking:=true;
End;
If checking=false then
begin
   clrscr;
   writeln; writeln; writeln;
   writeln('Course Not Found !!!!!!!!!!':53);
   writeln; writeln;
   delay(1000);
   End;
write('Do you want to delete more courses (Y/N) : ');
readln(Choose);
write(chr(7));
end; { End of While }
************
(This procedure helps start a new student record)
(Asks for a SS#. Checks if it is a valid number)

Procedure Adddata(Var counter: integer);

Var find: boolean;
Var first, second: string[10];
Var adder: char;

Begin
clrscr;
writeln;
writeln;
Counter:=Counter+1;
second:="";
write('Please input the social security # of new student: ');
readln(first);
if length(first) <> 9 then 
  (Length of SS# should be equal to nine)
Begin
  find:=false;
  while not(find) do
  begin
    clrscr;
    writeln;
    writeln('Social Security number should be exactly "NINE" digits long.');
    writeln;
    write('Please input the social security # of new student: ');
    readln(first);
    if length(first) <> 9 then find:=false
    else find:=true;
  end;
for I:= 1 to 8 do
  Begin
    adder:=first[I];
    second:=second+adder;
  End;
second:=second+'. '+first[9];
Datarray[Counter]:=second;
write(chr(7));
End;

************
(This procedure writes and saves all the SS#’s in a file called Master.Dat)

Procedure WriteData(Counter: integer);

Begin
rewrite(Datafile);

for I:= 1 to counter do
  writeln(Datafile,Datarray[I]);
Procedure Addinfo;

Var Ch,Check1,Check2,Select : Char;
Var Selection : Boolean;

Begin
   rewrite(Infofile);
   clrscr;
   writeln;
   write('Please Input the name of the Student : ':'50);
   readln(Name);
   if name=chr(27) then printmenu(Choice);
   writeln;
   write(chr(7) ;
   write('please Input the name of the Advisor , : 50) ;
   readln(Advisor);
   writeln;
   write(chr(7));
   writeln('Please Input the name of the Major : '50);
   writeln('(Examples of Majors : MATH, CS, REL) '48);
   GotoXY(51,7);
   readln(Major) ;
   writeln;
   writeln('
   writeln('Please Input the name of the School : '50);
   writeln('(Examples of Schools: Liberal Arts) '50);
   GotoXY(51,10);
   readln(School);
   writeln;
   writeln;
   writeln('Please Input the name of the Degree : '50);
   writeln('(Examples of Degrees: BS, BA, BFA) '50);
   GotoXY(51,13);
   readln(Degree);
   writeln;
   write('Are all the inputs correct (Y/N) :
   readln(check1);

   {Checks if all the inputs are correct. If not it calls the edit module)
   
   while check1 in ['n', 'N'] do
      Begin
         request1:=true;  {Only personal data edit will activated}
         request2:=false; {The course edit will not be activated}
         editrecord;
         writeln;
         write('Are all the inputs correct (Y/N) :
         readln(check1);
      End;

      {If all inputs are correct it writes them in a file which has the name }
      {as the students SS#}
      if check1 in ['y', 'Y'] then
Begin

crscr;
write(chr(7));
writeln(Infofile,Name);
writeln(Infofile,Advisor);
writeln(Infofile,Major);
writeln(Infofile,School);
writeln(Infofile,Degree);
writeln; writeln;
write('Do you want to enter courses (Y/N) : ':55);
readln(Ch);
write(chr(7));
End
else ch:='n';

{ This module helps the user enter a students courses }
while ((ch = 'y') or (ch = 'Y')) do

begin (Begin of While)

crscr;
writeln; writeln;
writeln('Input Department ':45);
writeln('(Note: All Uppercase) ':45);
GotoXY(46,3);
readln(Course.dept);
writeln;
write('Input Course Number ':45);
readln(Course.Number);
writeln;
write('Input Course Unit ':45);
readln(Course.Value);
writeln;
write('Input Course Grade ':45);
readln(Course.Grade);
writeln;
writeln;
write('Are all the inputs correct (Y/N) : ');
readln(check2); {Checks if all the info about a particular course is okay}

select:=' ';

{ If the information needs to be editted then check2 is 'N' }
While check2 in ['n','N'] do

Begin
While select<>'5' do
Begin
selection:=false;
crscr;
write(chr(7));
crscr;
writeln; writeln;
writeln('Please input what you want to edit ':54);
writeln('---------------------------------- ':54);
writeln; writeln;
write('Please input what you want to edit ':54);
write('---------------------------------- ':54);
writeln; writeln;
writeln;
write("1": Course Dept : ":42); writeln(Course.Dept);
write("2": Course Number : ":42); writeln(Course.Number);
write("3": Course Unit : ":42); writeln(Course.Value);
write("4": Course Grade : ":42); writeln(Course.Grade);
write("5": Done ":42);
writeln; writeln;
write('Input Choice , : 42);
readln(Select);
write(chr(7));

{ Helps determine which part of the record needs to be edited }

Case select of

'1': Begin
writeln;
write('Please input the edited Course Dept : '); 
readln(Course.Dept);
write(chr(7));
selection:=true;
End;

'2': Begin
writeln;
write('Please input the edited Course Number : ');
readln(Course.Number);
write(chr(7));
selection:=true;
End;

'3': Begin
writeln;
write('Please input the edited Course Unit : ');
readln(Course.Value);
write(chr(7));
selection:=true;
End;

'4': Begin
writeln;
write('Please input the edited Course Grade : ');
readln(Course.Grade);
write(chr(7));
selection:=true;
End;

End; {End of Case}

If select='5' then selection:=true;

If selection=false then 
Begin
clrscr;
writeln; writeln;
write(chr(7)); Choice Not Found!!!!!!!'

End;

If selection=true then


Begin
    writeln; writeln;
    writeln('Are all the inputs correct (Y/N) : '); readln(check2);
    If check2 in ['y','Y'] then select:= '5';
End;

End; (End of Select)
End; (End of Editing)

(If everything is okay the students information is written to Infofile)

write(chr(7));
if check2 in ['y','Y'] then
    Begin
        writeln(Infofile,Course.Dept);
        writeln(Infofile,Course.Number);
        writeln(Infofile,Course.Value);
        writeln(Infofile,Course.Grade);
    End;
    writeln;
    writeln('Do you want to enter more courses (Y/N) : ');
    readln(Ch);
    write(chr(7));
end; ( End of enter courses )

close(infofile);

End;

*************************************************************************
( This procedure helps in getting a student record from the c:\directory )
( The name of the file is the same as the students SS#)

Procedure GetRecord;

Begin
    clrscr;
    count:=1;
    check:=false;
    writeln; writeln;
    writeln('Please input the SS# of the student record you want: ');
    readln(Num);
    Eight:=Num;
    Num:=Eight+'.'+Num[9];
    write(chr(7));
    while ((Num<>datarray[Count]) and (Count<Counter)) do
        Begin
            count:=count+1;
        End;

    if Num=datarray[count] then
        check:=true
    else
        begin
            writeln; writeln;
        End;

End;
writeln('Record Not Found !!!!!!!!':55);
write(chr(7),chr(7),chr(7));
Gotoxy(1,18);
write('Press Any Key To Continue............':77);
while not(keypressed) do;
readln;
end;
End;
{*************************************************************************}
{This procedure reads in all the information about the record requested}
{into appropriate arrays}
Procedure ReadRecord;
Begin
  reset(Infofile);
  readline(Infofile,Name);
  readline(Infofile,Advisor);
  readline(Infofile,Major);
  readline(Infofile,School);
  readline(Infofile,Degree);
Counting:=0;
coursearray[Counting].Dept:='';
While (Coursearray[Counting].Dept <> '') do
  begin
    counting:=counting+1;
    readline(Infofile,Coursearray[Counting].Dept);
    readline(Infofile,Coursearray[Counting].Number);
    readline(Infofile,Coursearray[Counting].Value);
    readline(Infofile,Coursearray[Counting].Grade);
  if ((coursearray[Counting].Number=0) and
      (coursearray[Counting].Grade='') )
    then
      counting:=counting-1;
  end;
  counting:=counting-1;
End;
{*************************************************************************}
{ Before exiting the current record, this module helps write all the }
{ information back into the students file }
Procedure WriteRecord;
Begin
  rewrite(Infofile);
  writeln(Infofile, Name);
  writeln(Infofile, Advisor);
  writeln(Infofile, Major);
  writeln(Infofile, School);
  writeln(Infofile, Degree);
  For I:= 1 to counting do  {counting is the # of courses the student has}
    begin
      writeln(Infofile,Coursearray[I].Dept);
writeln(Infofile,Coursearray[I].Number);
writeln(Infofile,Coursearray[I].Value);
writeln(Infofile,Coursearray[I].Grade);
End;

close(Infofile);
End;

{******************************************************************************
{This procedure processes the following information : }
{   1. The number of D grades in major/majors }
{   2. The number of 300-400 level courses    }
{   3. The number of total units }

Procedure ProcessRecord;
Var Majcount : Integer;
Var Tmajor1, Tmajor2 : string[10];

Begin
  Higher:=0;
  Dmajor:=0;
  Dmajor2:=0;
  Dunit:=0;
  Tunit:=0;
  Tmajor1:='';
  Tmajor2:='';
  Mcheck:=false;

  { Checks to see if a student has two majors }
  For I:= 1 to length(major) do
    If major[i]='/') then
      Begin
        Majcount:=I;
        Mcheck:=true;
      End;
  If Mcheck=true then
    Begin
      For I:= 1 to (Majcount-1) do
        Tmajor1:=Tmajor1+major[I];
    End;
  If Mcheck=true then
    Begin
      For I:= (Majcount+1) to length(major) do
        Tmajor2:=Tmajor2+major[I];
      Major:=Tmajor1;
      Major2:=Tmajor2;
    End;

  For I:=1 to counting do
    Begin
      val(coursearray[I].Value, Adder, Code);
      If ((coursearray[I].Grade='F') or (coursearray[I].Grade='NCR')
        or (coursearray[I].Grade = 'IN') or (coursearray[I].Grade = 'DR'))
        then Adder:=0;
      If ((coursearray[I].Number>299) and (coursearray[I].Grade <> 'F'))
and (coursearray[I].Grade <> 'IN') and (coursearray[I].Grade <> 'DR')
and (coursearray[I].Grade <> 'W') and (coursearray[I].Grade <> 'P')
and (coursearray[I].Grade <> 'C') and (coursearray[I].Grade <> 'NCR')
and (Adder > 0.69)
then Higher:=Higher+Adder;
If coursearray[I].Grade='D' then
  Dunit:=Dunit+Adder;
  Tunit:=Tunit+Adder;
If ((coursearray[I].Dept=Major) and (coursearray[I].Grade='D'))
then DMajor:=Dmajor+Adder;
If ((coursearray[I].Dept=Major2) and (coursearray[I].Grade='D'))
then DMajor2:=Dmajor2+Adder;
End;
End;

{*************************************************************************
{ This procedure performs the humanities checks }
{ Criterion this procedure uses is that grade should not be }
{ W, IN, CR, NCR, P or F; the unit value of the course should be > 0.69 }
{ All the procedures that have the prefix process work in the similar way}
{ They assign the variable the course name and number if all the criterion }
{ is met. }

Procedure ProcessHum;
Var Temp:integer;
Var LCheck, PhCheck, Hcheck: Boolean;
Begin
  Temp:=0;
  Hcount:=1;
  Lcheck:=true; Phcheck:=true; Hcheck:=true;
  While Temp<counting do
  Begin
    Temp:=Temp+1;
    val(coursearray[Temp].Value, Adder, Code);
    If ((coursearray[Temp].Dept='ENGL') and (Litl.Numbers=0) and (HCount<3)
and (coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'P')
and (coursearray[Temp].Number > 269) and (coursearray[Temp].Number <> 30
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <> 'N'
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'D'
then
  Begin
    Hcount:=Hcount+1;
    Lcheck:=false;
    LITL.Depts:=coursearray[Temp].Dept;
    LITL.Numbers:=coursearray[Temp].Number;
  End;
  If (((coursearray[Temp].Dept='FREN') or (coursearray[Temp].Dept='GER')
or (coursearray[Temp].Dept='GRK') or (coursearray[Temp].Dept='JAPN')
or (coursearray[Temp].Dept='RUSS') or (coursearray[Temp].Dept='SPAN'))
and ((coursearray[Temp].Number=307) or (coursearray[Temp].Number=308) or
(coursearray[Temp].Number=408) or (coursearray[Temp].Number=377))
and (LITL.Numbers=0) and (HCount<3) and (Lcheck=true)
and (coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'P')
and (coursearray[Temp].Grade <> 'C') and (coursearray[Temp].Grade <> 'N'
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'F') and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'IN') then Begin
Hcount:=Hcount+1;
Lcheck:=false;
Lit1.Depts:=coursearray[Temp].Dept;
Lit1.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='PHIL') and (Phil1.Numbers=0) and (HCount<3) and (coursearray[Temp].Grade <> 'F') and (Phcheck=true) and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <> 'IN') then Begin
Hcount:=Hcount+1;
Phcheck:=false;
Phil1.Depts:=coursearray[Temp].Dept;
Phil1.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='HUM') and (Hum1.Numbers=0) and (HCount<3) and (coursearray[Temp].Grade <> 'F') and (Hcheck=true) and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <> 'IN') then Begin
Hcount:=Hcount+1;
Hcheck:=false;
Hum1.Depts:=coursearray[Temp].Dept;
Hum1.Numbers:=coursearray[Temp].Number;
End;

************
{ This procedure performs the Social Science checks }

Procedure ProcessSS;

Var Temp:integer;

Begin
Temp:=0;
SScount:=1; (Calculates how many you need. For example you could need three courses from a possible of five)

While Temp<Counting do Begin
Temp:=Temp+1;
val(coursearray[Temp].Value, Adder, Code);
If ((coursearray[Temp].Dept='ECON') and (Econ.Numbers=0) and (SScount<3) and (coursearray[Temp].Grade <> 'F') and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <> 'IN') then Begin
Hcount:=Hcount+1;
...
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <>
then
Begin
SScount := SScount + 1;
Econ.Depts := coursearray[Temp].Dept;
Econ.Numbers := coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='HIST') and (Hist.Numbers=0) and (SScount<3)
and ((coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <>
then
Begin
SScount := SScount + 1;
Hist.Depts := coursearray[Temp].Dept;
Hist.Numbers := coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='PSCI') and (PSCI.Numbers=0) and (SScount<3)
and ((coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <>
then
Begin
SScount := SScount + 1;
PSCI.Depts := coursearray[Temp].Dept;
PSCI.Numbers := coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='SOC') and (Soc.Numbers=0) and (SScount<3)
and ((coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <>
then
Begin
SScount := SScount + 1;
Soc.Depts := coursearray[Temp].Dept;
Soc.Numbers := coursearray[Temp].Number;
End;

If degree <> 'BSN' then
If (((coursearray[Temp].Dept='FREN') or (coursearray[Temp].Dept='GER')
or (coursearray[Temp].Dept='GRK') or (coursearray[Temp].Dept='JAPN')
or (coursearray[Temp].Dept='RUSS') or (coursearray[Temp].Dept='SPAN')
and (coursearray[Temp].Number=317))
and (sosc.Numbers=0) and (SScount<3)
and ((coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <>
then
Begin
SScount:=SScount+1;
sosc.Depts:=coursearray[Temp].Dept;
sosc.Numbers:=coursearray[Temp].Number;
End;

If Degree='BSN' then
  If ((coursearray[Temp].Dept='SOSC') and (coursearray[Temp].Number=102)
  and (coursearray[Temp].Grade <> 'F')
  and (coursearray[Temp].Grade <> 'W')
  and (coursearray[Temp].Grade <> 'CR')
  and (coursearray[Temp].Grade <> 'IN')
  and (Adder > 0.69)
  then
    Begin
      SS102.Depts:=coursearray[Temp].Dept;
      SS102.Numbers:=coursearray[Temp].Number;
    End;
  End;
End;

******************************************************************************

(This procedure processes the Religion, Expository Writing checks)

Procedure ProcessOthers;
Var Temp: integer;
Begin
  Temp:=0;
  While Temp<Counting do
    Begin
      Temp:=Temp+1;
      val(coursearray[Temp].Value, Adder, Code);
      If (((coursearray[Temp].Dept='ENGL') or (coursearray[Temp].Dept='FS'))
      and (Eng.Numbers=0) and ((coursearray[Temp].Grade <> 'F')
      and (coursearray[Temp].Grade <> 'W')
      and (coursearray[Temp].Grade <> 'CR')
      and (coursearray[Temp].Grade <> 'IN')
      and (coursearray[Temp].Number < 106)
      )
      then
        Begin
          Eng.Depts:=coursearray[Temp].Dept;
          Eng.Numbers:=coursearray[Temp].Number;
        End;
      If ((coursearray[Temp].Dept='REL') and (Rel.Numbers=0) and
      ((coursearray[Temp].Grade <> 'F')
      and (coursearray[Temp].Grade <> 'W')
      and (coursearray[Temp].Grade <> 'CR')
      and (coursearray[Temp].Grade <> 'IN')
      and (Adder > 0.69)
      and (coursearray[Temp].Number < 106)
      )
      then
        Begin
          Rel.Depts:=coursearray[Temp].Dept;
          Rel.Numbers:=coursearray[Temp].Number;
        End;
    End;
End; {End of While}

End;

*************************************************************************
{ This procedure processes the Foreign Language checks }

Procedure ProcessOthers2;

Var Temp: integer;
Var Fcheck: boolean;

Begin
  Temp:=0;

While Temp<Counting do
  Begin
    Fcheck:=true;
    Temp:=Temp+1;
    val(coursearray[Temp].Value, Adder, Code);
    If ((coursearray[Temp].Dept='FREN') or (coursearray[Temp].Dept='GER')
      or (coursearray[Temp].Dept='GRK') or (coursearray[Temp].Dept='JAPN')
      or (coursearray[Temp].Dept='RUSS') or (coursearray[Temp].Dept='SPAN')
      ) and (coursearray[Temp].Number=201)
      and (coursearray[Temp].Grade <> 'F')
      and (coursearray[Temp].Grade <> 'IN')
      and (coursearray[Temp].Grade <> 'D')
      and (coursearray[Temp].Grade <> 'CR')
      and (coursearray[Temp].Grade <> 'W')
      )
    then
      Begin
        Flang.Depts:=coursearray[Temp].Dept;
        Flang.Numbers:=coursearray[Temp].Number;
      End;

    If ((coursearray[Temp].Dept='FREN') or (coursearray[Temp].Dept='GER')
      or (coursearray[Temp].Dept='GRK') or (coursearray[Temp].Dept='JAPN')
      or (coursearray[Temp].Dept='RUSS') or (coursearray[Temp].Dept='SPAN')
      ) and (Flang2.Numbers=0)
      and (Flang2.Numbers<>0) and (coursearray[Temp].Grade <> 'F')
      and (coursearray[Temp].Grade <> 'IN')
      and (coursearray[Temp].Grade <> 'D')
      and (coursearray[Temp].Grade <> 'CR')
      and (coursearray[Temp].Grade <> 'W')
      )
    then
      Begin
        Flang2.Depts:=coursearray[Temp].Dept;
        Flang2.Numbers:=coursearray[Temp].Number;
        Fcheck:=False;
      End;

    If ((coursearray[Temp].Dept='FREN') or (coursearray[Temp].Dept='GER')
      or (coursearray[Temp].Dept='GRK') or (coursearray[Temp].Dept='JAPN')
      or (coursearray[Temp].Dept='RUSS') or (coursearray[Temp].Dept='SPAN')
      ) and (Flang3_Numbers=0) and (Fcheck=true)
      and (Flang3.Numbers<>0)
      and (Flang2.Numbers <> 0)
      and (Fcheck)}
((coursearray[Temp].Grade <> 'F')
and((coursearray[Temp].Grade <> 'IN')
and((coursearray[Temp].Grade <> 'CR')
and((coursearray[Temp].Grade <> 'W'))))
then
Begin
  Flang3.Depts:=coursearray[Temp].Dept;
  Flang3.Numbers:=coursearray[Temp].Number;
  Fcheck:=False;
End;
End; (End of While)
End;

(*This procedure performs the Natural Science checks. The common parts are handled in this procedure*)

Procedure ProcessNS;

Var Temp:integer;

Begin
  NScount:=0; {Counts up till all the Natural Science Requirements are met}
  Temp:=0;
  While Temp<Counting do
    Begin
      Temp:=Temp+1;
      val(coursearray[Temp].Value, Adder, Code);
      If (((coursearray[Temp].Dept='MATH') or (coursearray[Temp].Dept='CS'))
      and (MaCs.Numbers=0) and (coursearray[Temp].Grade <> 'F')
      and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade
      and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade
      and (Adder > 0.69)
      and (coursearray[Temp].Grade <> 'W')) then
        Begin
          MaCs.Depts:=coursearray[Temp].Dept;
          MaCs.Numbers:=coursearray[Temp].Number;
          NScount:=NScount+1;
        End;
      If (((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=101))
      or ((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=102))
      or ((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=105))
      or ((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=106))
      or ((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=207))
      or ((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=110))
      or ((coursearray[Temp].Dept='PHYS') and (coursearray[Temp].Number=120))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=101))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=102))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=101))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=102))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=104))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=110))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=120))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=130))
      or ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=311))
      or ((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=101))
((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=102)) or
((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=104)) or
((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=107)) or
((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=108)) or
((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=210)) or
((coursearray[Temp].Dept='PSYC') and (coursearray[Temp].Number=211)) or
((coursearray[Temp].Dept='PSYC') and (coursearray[Temp].Number=212)) and
(LabSc.Numbers=0) and ((coursearray[Temp].Grade <> 'F') and
(coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade < 'CR') and (coursearray[Temp].Grade < 'CR') and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'IN')
then
Begin
If LabSc.Numbers=0 then
Begin
LabSc.Depts:=coursearray[Temp].Dept;
LabSc.Numbers:=coursearray[Temp].Number;
End;
End;
End;

{*******************************************************************************}
{ This procedure performs the Natural Science checks. All special cases are handled here }
Procedure ProcessNS2;
Var Temp :integer;
Var Nscheck : Boolean;

Begin
Temp:=0;
NScount1:=0;

While Temp<counting do
Begin
Temp:=Temp+1;
Nscheck:=true;
val(coursearray[Temp].Value, Adder, Code);
If Degree='BME' then
Begin
If (((coursearray[Temp].Dept='MATH') and (Math.Numbers=0) and (coursearray[Temp].Grade <> 'F') and
(coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade < 'CR') and (coursearray[Temp].Grade < 'CR') and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'IN')
then
Begin
Math.Numbers:=coursearray[Temp].Number;
End;
End;
If (((coursearray[Temp].Dept='BIOL') or (coursearray[Temp].Dept='PSYC')
or ((coursearray[Temp].Dept='NASC') and (coursearray[Temp].Number=1 and
(Life.Numbers=0) and (coursearray[Temp].Grade <> 'F') and
(coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade < 'CR') and (coursearray[Temp].Grade < 'CR') and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <
and (NScount<2) ))
then
Begin
If ((NScount<2) AND (coursearray[Temp].Grade>'IN')) then
Begin
Life.Depts:=coursearray[Temp].Dept;
Life.Numbers:=coursearray[Temp].Number;
NScount:=NScount+1;
End;
End;

If ((coursearray[Temp].Dept='CHEM')
and (PhSc.Numbers=0) and (coursearray[Temp].Grade <> 'F' ) and (NSco
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder> 0.69)
and (coursearray[Temp].Grade <> 'IN')and(coursearray[Temp].Grade <> ')
then
Begin
PhSc.Depts:=coursearray[Temp].Dept;
PhSc.Numbers:=coursearray[Temp].Number;
NScount:=NScount+1;
End;

If ((coursearray[Temp].Dept='PHYS')
and (PhSc.Numbers=0) and (coursearray[Temp].Grade <> 'F' ) and (NSco
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder> 0.69)
and (coursearray[Temp].Grade <> 'IN')and(coursearray[Temp].Grade <> ')
then
Begin
PhSc.Depts:=coursearray[Temp].Dept;
PhSc.Numbers:=coursearray[Temp].Number;
NScount:=NScount+1;
End;

If ((coursearray[Temp].Dept='GEOL') AND (coursearray[Temp].Number=101)
and (PhSc.Numbers=0) and (coursearray[Temp].Grade <> 'F' ) and (NSco
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder> 0.69)
and (coursearray[Temp].Grade <> 'IN')and(coursearray[Temp].Grade <> ')
then
Begin
PhSc.Depts:=coursearray[Temp].Dept;
PhSc.Numbers:=coursearray[Temp].Number;
NScount:=NScount+1;
End;

If (Degree='BME') then
Begin
If (((coursearray[Temp].Dept='CHEM') or (coursearray[Temp].Dept='PHYS')
or (coursearray[Temp].Dept='BIOL'))
and (NS1.Numbers=0) and (NScount1<3)
and (coursearray[Temp].Grade <> 'F' ) and (NScheck=true)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder> 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade
then
  Begin
    NScount1:=NScount1+1;
    NScheck:=false;
    NS1.Depts:=coursearray[Temp].Dept;
    NS1.Numbers:=coursearray[Temp].Number;
  End;

If (((coursearray[Temp].Dept='CHEM') or (coursearray[Temp].Dept='PHYS')
  or (coursearray[Temp].Dept='BIOL'))
  and (NS2.Numbers=0) and (NScount1<3)
  and (coursearray[Temp].Grade <> 'F') and (NScheck=true)
  and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <
  and (coursearray[Temp].Grade <> 'CR') and (Adder > 0.69)
  and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade

then
  Begin
    NScount1:=NScount1+1;
    NScheck:=false;
    NS2.Depts:=coursearray[Temp].Dept;
    NS2.Numbers:=coursearray[Temp].Number;
  End;

If (((coursearray[Temp].Dept='CHEM') or (coursearray[Temp].Dept='PHYS')
  or (coursearray[Temp].Dept='BIOL'))
  and (NS3.Numbers=0) and (NScount1<3)
  and (coursearray[Temp].Grade <> 'F') and (NScheck=true)
  and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <
  and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade
  and (Adder > 0.69)
  and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade

then
  Begin
    NScount1:=NScount1+1;
    NScheck:=false;
    NS3.Depts:=coursearray[Temp].Dept;
    NS3.Numbers:=coursearray[Temp].Number;
  End;

End;

End;

{*************************************************************************}
{ This procedure performs the special requirements in BSN }

Procedure ProcessBSN;

Var Temp : integer;

Begin
  Temp:=0;

While Temp<Counting do
  Begin
    Temp:=Temp+1;
    val(coursearray[Temp].Value, Adder, Code);

    If ((coursearray[Temp].Dept='PHIL') and (coursearray[Temp].Number=102)
and (coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
then
Begin
Log.Depts:=coursearray[Temp].Dept;
Log.Numbers:=coursearray[Temp].Number;
End;

If ((((coursearray[Temp].Dept='NASC') and (coursearray[Temp].Number=101))
and (coursearray[Temp].Grade <> 'F'))
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <
then
Begin
If (coursearray[Temp].Grade<>'IN') then
Begin
NSI01.Depts:=coursearray[Temp].Dept;
NSI01.Numbers:=coursearray[Temp].Number;
End;
End;

If ((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=107)
and (coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <=
then
Begin
Biol07.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='BIOL') and (coursearray[Temp].Number=108)
and (coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <=
then
Begin
Biol08.Depts:=coursearray[Temp].Dept;
Biol08.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='CHEM') and (coursearray[Temp].Number=110)
and (coursearray[Temp].Grade <> 'F')
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <=
then
Begin
Chem110.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='PHIL') and (coursearray[Temp].Number=103) or (coursearray[Temp].Number=104) or (coursearray[Temp].Number=109) or (coursearray[Temp].Number=110) or (coursearray[Temp].Number=111) or (coursearray[Temp].Number=120) or (coursearray[Temp].Number=270) or (coursearray[Temp].Number=271) or (coursearray[Temp].Number=112)) and (coursearray[Temp].Grade <> 'F') and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grade <> 'F')
then
Begin
PhilN.Depts:=coursearray[Temp].Dept;
PhilN.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='HUM') or (coursearray[Temp].Dept='ENGL') and (coursearray[Temp].Number>269) and (HumLit.Numbers=0) and (coursearray[Temp].Grade <> 'F') and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grade < 'IN')
then
Begin
HumLit.Depts:=coursearray[Temp].Dept;
HumLit.Numbers:=coursearray[Temp].Number;
End;
End;

{**************************************************************************}
{ This procedure processes the course that is an elective }
Procedure ProcessElec;
Var Temp:integer;
Var Echeck:Boolean;
Begin
Temp:=0;
While ((Temp<Counting) and (Elec.Numbers=0)) do
Begin
Temp:=Temp+1;
val(coursearray[Temp].Value, Adder, Code);
Echeck:=True;
If (Elec.Numbers=0) and (coursearray[Temp].Dept<>'MUS') and (coursearray[Temp].Grade <> 'F') and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade <> 'CR')
then
End;
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grade <

Begin

If ((coursearray[Temp].Dept=Eng.Depts) and (coursearray[Temp].Number=Eng.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Flang.Depts) and (coursearray[Temp].Number=Flang.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Flang2.Depts) and (coursearray[Temp].Number=Flang2.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Rel.Depts) and (coursearray[Temp].Number=Rel.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Hum1.Depts) and (coursearray[Temp].Number=Hum1.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Lit1.Depts) and (coursearray[Temp].Number=Lit1.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Phil1.Depts) and (coursearray[Temp].Number=Phil1.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Econ.Depts) and (coursearray[Temp].Number=Econ.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Hist.Depts) and (coursearray[Temp].Number=Hist.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Psci.Depts) and (coursearray[Temp].Number=Psci.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=MaCS.Depts) and (coursearray[Temp].Number=Macs.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=Soc.Depts) and (coursearray[Temp].Number=Soc.Number) then Echeck:=False;
If ((coursearray[Temp].Dept=sosc.Depts) and (coursearray[Temp].Number=sosc.Number) then Echeck:=False;
If Echeck=True
then
Begin
Elec.Depts:=coursearray[Temp].Dept;
Elec.Number:=coursearray[Temp].Number;
End;
End; {End of While}
End;

*************************************************************************
{ This procedure performs the Fine Arts checks }
Procedure ProcessFA;

Var Temp:integer;

Begin
Temp:=0;

While Temp<Counting do
Begin
val(coursearray[Temp].Value, Adder, Code);
Temp:=Temp+1;
If (((coursearray[Temp].Dept='ART') and (coursearray[Temp].Number=111)) or 
((coursearray[Temp].Dept='ART') and (coursearray[Temp].Number=113)) or 
((coursearray[Temp].Dept='ART') and (coursearray[Temp].Number=115)) or 
((coursearray[Temp].Dept='ART') and (coursearray[Temp].Number=116)) or 
((coursearray[Temp].Dept='ART') and (coursearray[Temp].Number=130)) or 
((coursearray[Temp].Dept='ART') and (coursearray[Temp].Number=135)) or
begin
Temp:=0;
PCount:=0;

while Temp<Counting do
begin
Temp:=Temp+1;
val(coursarray[Temp].Value, Adder, Code);
Pcheck:=true;
if ((coursarray[Temp].Dept='PEC') and (Pecl.Numbers=0) and (PCount<4)
and (coursarray[Temp].Grade <> 'NCR') and (Pcheck=true)
and (Fart.Numbers=0) and (coursarray[Temp].Grade <> 'F')
and (coursarray[Temp].Grade <> 'W') and (coursarray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <>
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'D'
then
begin
Fart.Depts:=coursarray[Temp].Dept;
Fart.Numbers:=coursarray[Temp].Number;
end;
{ Music 200 is a psuedo coursenumber. It is input if the FA requirements
has been satified by Music 0.25 credit classes }
end;
end;

*************************************************************************
{ This procedure performs Physical Education checks }

procedure ProcessPE;

var Temp:integer;
var PCheck:boolean;

begin
Temp:=0;
PCount:=0;

while Temp<Counting do
begin
Temp:=Temp+1;
val(coursarray[Temp].Value, Adder, Code);
Pcheck:=true;
if ((coursarray[Temp].Dept='PEC') and (Pecl.Numbers=0) and (PCount<4)
and (coursarray[Temp].Grade <> 'NCR') and (Pcheck=true)
and (Fart.Numbers=0) and (coursarray[Temp].Grade <> 'F')
and (coursarray[Temp].Grade <> 'W') and (coursarray[Temp].Grade <>
and (coursearray[Temp].Grade <> 'CR') and (coursearray[Temp].Grade <>
and (Adder > 0.69)
and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade <> 'D'
then
begin
Fart.Depts:=coursarray[Temp].Dept;
Fart.Numbers:=coursarray[Temp].Number;
end;
{ Music 200 is a psuedo coursenumber. It is input if the FA requirements
has been satified by Music 0.25 credit classes }
end;
end;
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grade <> 'IN') and (coursearray[Temp].Grade < 'IN')
then
Begin
If coursearray[Temp].Value='X' then
  Pecount:=Pecount+2
else Pecount:=Pecount+1;
Pcheck:=false;
Pec1.Depts:=coursearray[Temp].Dept;
Pec1.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='PEC') and (Pec2.Numbers=0) and (PeCount<4)
and (coursearray[Temp].Grade <> 'NCR') and (Pcheck=true)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grad
then
Begin
If coursearray[Temp].Value='X' then
  Pecount:=Pecount+2
else Pecount:=Pecount+1;
Pcheck:=false;
Pec2.Depts:=coursearray[Temp].Dept;
Pec2.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='PEC') and (Pec3.Numbers=0) and (PeCount<4)
and (coursearray[Temp].Grade <> 'NCR') and (Pcheck=true)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grad
then
Begin
If coursearray[Temp].Value='X' then
  Pecount:=Pecount+2
else Pecount:=Pecount+1;
Pcheck:=false;
Pec3.Depts:=coursearray[Temp].Dept;
Pec3.Numbers:=coursearray[Temp].Number;
End;

If ((coursearray[Temp].Dept='PEC') and (Pec4.Numbers=0) and (PeCount<4)
and (coursearray[Temp].Grade <> 'NCR') and (Pcheck=true)
and (coursearray[Temp].Grade <> 'W') and (coursearray[Temp].Grade < 'IN') and (coursearray[Temp].Grad
then
Begin
If coursearray[Temp].Value='X' then
  Pecount:=Pecount+2
else Pecount:=Pecount+1;
Pec4.Depts:=coursearray[Temp].Dept;
Pec4.Numbers:=coursearray[Temp].Number;
End;
End;

*****************************************************************************
{ BFA, BM, BME }  
{ This procedure prints the results after a record has been processed }  
{ onto the screen, in the format of the form }
Procedure WRecord2;

Var Total1, D1 : integer; {Stores different requirements for the different Degrees}

Begin

clrscr;
If degree='BFA' then total1:=34 else if
  degree='BM' then total1:=37 else if
  degree='BME' then total1:=39;

If degree='BFA' then D1:=4 else if
  degree='BM' then D1:=6 else if
  degree='BME' then D1:=4;

clrscr;
IEWLLINOIS WESLEYAN UNIVERSITY':53);
writeln('':28,Degree,' CREDIT CHECK FORM ');
writeln;
writeln;
write(' Student Name : ',Name);
  For i:=length(Name) to 25 do
    write(' '); 
writeln('Advisor Name : ',Advisor);
write(' Major1 : ',Major);
  For i:=length(Major) to 25 do
    write(' '); 
write('Major2 : ',Major2);
writeln; writeln;
writeln('GENERAL UNIVERSITY REQUIREMENTS':54);
writeln; writeln;
If Eng.Depts='' then write('________')
  else
    write(' O.K. ');
write(' I. Writing: English 105 or equivalent: ');
If Eng.Depts<>'' then write(Eng.Depts,' ',Eng.Numbers)
  else write('________');
writeln; writeln;
If Flang2.Depts='' then write('________')
  else
    write(' O.K. ');
write(' II. Foreign Language: ');
If Flang.Depts<>'' then write(Flang.Depts,' ',Flang.Numbers)
  else write('________');
If Flang2.Depts<>'' then write(' ',Flang2.Depts,' ',Flang2.Numbers)
  else write('________');
If degree='BME' then
  else write('________');
writeln; writeln;
If Rel.Depts='' then write('________')
  else
    write(' O.K. ');
write(' III. Religion: ');
If Rel.Depts<>'' then write(Rel.Depts,' ',Rel.Numbers)
  else write('________');
If Hcount>2 then
  writeln(' O.K. ', ' IV. Humanities: ');
else writeln(' ',' IV. Humanities: ');

writeln;'writeln;
writeln('Two courses from at least two of the following':61);
writeln('areas:':21);
delay(3000); clrscr;
write(' 1. Literature ');
If Lit1.Depts<>'' then writeln(','Lit1.Depts,, ',Lit1.Numbers)
else writeln(' ______ ');
write(' 2. Philosophy ');
If Phil1.Depts<>'' then writeln(','Phil1.Depts,, ',Phil1.Numbers)
else writeln(' ______ ');
write(' 3. Humanities ');
If Hum1.Depts<>'' then writeln(','Hum1.Depts,, ',Hum1.Numbers)
else writeln(' ______ ');
writeln;
writeln;
If SScount=3 then
  writeln(' O.K. V. Social Science :')
else writeln(' ______ V. Social Science :');
writeln;
writeln(' Two courses from two different areas:');
writeln;
write('Economics: ');
If Econ.Depts<>'' then write(','Econ.Depts,, ',Econ.Numbers)
else write(' ______ ');
write('History: ');
If Hist.Depts<>'' then write(','Hist.Depts,, ',Hist.Numbers)
else write(' ______ ');
write('Political Science: ');
If Psci.Depts<>'' then write(','Psci.Depts,, ',Psci.Numbers)
else write(' ______ ');
write('Sociology (not 291): ');
If Soc.Depts<>'' then write(','Soc.Depts,, ',Soc.Numbers)
else write(' ______ ');
write('Social Science: ');
If sosc.Depts<>'' then write(','sosc.Depts,, ',sosc.Numbers)
else write(' ______ ');
writeln;
writeln;
If ((Degree='BM') or (Degree='BFA')) then
Begin
  If (NScount>1)
  then
    Begin
      writeln(' O.K. VI. Natural Science: at least two course units from at
      writeln(' two of the following groups');
    End
  else
    Begin
      writeln(' VI. Natural Science: at least two course units from')
    End
End

writeln(' two of the following groups');
End;
writeln;
write(' Group 1: (biology, psychology, NatSci 101 ) ];
If Life.Depts<>'' then writeln(Life.Depts,' ',Life.Numbers)
else writeln('_____');
write(' Group 2: (chemistry, physics ) ');
If PhSc.Depts<>'' then writeln(PhSc.Depts,' ',PhSc.Numbers)
else writeln('_____');
write(' Group 3: (mathematics, computer science ) ');
If MaCs.Depts<>'' then writeln(MaCs.Depts,' ',MaCs.Numbers)
else writeln('_____');
writeln;
End;
If (Degree='BME') then
Begin
If ((NScount1>2) and (LabSc.Depts<>'')) then
   writeln(' O.K. VI. Natural Science: at least 3 course units from (biolog
else
   writeln('_____ VI. Natural Science: at least 3 course units from (biol
   writeln('_____ chemistry, or physics) and at least one laboratory
   writeln('_____ science: ');
writeln;
writeln;
If Labsc.Depts<>'' then write(' ',Labsc.Depts,' ',Labsc.Numbers)
else write('_____ ');
If NS1.Depts<>'' then write(' ',NS1.Depts,' ',NS1.Numbers)
else write('_____ ');
If NS2.Depts<>'' then write(' ',NS2.Depts,' ',NS2.Numbers)
else write('_____ ');
If NS3.Depts<>'' then write(' ',NS3.Depts,' ',NS3.Numbers)
else write('_____ ');
writeln; writeln;
End;
delay(3000);
clrscr;
writeln;
If Pecount>3 then
   writeln(' O.K. VII. Physical Education :')
else
   writeln('_____ VII. Physical Education :')
writeln;
writeln(' 2 courses (X) or 4 half courses (Y) or an equivalent:
write(' combination: ');}
If Pec1.Depts<>'' then write(' ',Pec1.Depts,' ',Pec1.Numbers)
else write('_____ ');
If Pec2.Depts<>'' then write(' ',Pec2.Depts,' ',Pec2.Numbers)
else write('_____ ');
If Pec3.Depts<>'' then write(' ',Pec3.Depts,' ',Pec3.Numbers)
else write('_____ ');
else writeln('_____ ');
writeln; writeln;
If degree='BM' then
Begin
  If Elec.Depts='' then write('_____')
  else
    write('  O.K. ');
  write(' VIII. Elective: 1 course unit non-music elective ');
  If Elec.Depts<>' then write(Elec.Depts, ' ', Elec.Numbers)
  else write('_____');
End;

If degree='BME' then
Begin
  If Math.Depts='' then write('_____')
  else
    write('  O.K. ');
  write(' VIII. Mathematics: one course unit ');
  else write('_____');
End;

writeln; writeln; writeln; writeln;
write(Higher:0:2);
If Higher<10 then write(' ');
writeln(' Total completed (11 course units numbered 300 and 400 are ':61);
writeln(' required with at least 4 in a departmental major)':74);
writeln;
write(Tunit:0:2);
If Tunit>9.99 then write(' ')
  else write(' ');
write(' Total units completed ( ',total1, ' course units required)');
writeln; writeln;
writeln(Dunit:0:2, ' Total "D" units (no more than ',D1,' will be counted tow
writeln; writeln;
write(Dmajor:0:2, ' Total "D" units in major1 : ',major);
writeln(' (no more than 1 will be counted ');
writeln(' toward graduation') );
writeln; writeln;
write(Dmajor2:0:2, ' Total "D" units in major2 : ',major2);
writeln(' (no more than 1 will be counted ');
writeln(' toward graduation') );
writeln;
while not(keypressed) do;
readln;
End;

*****************************************************************************
{ BFA,BM,BME to printer}
{ This procedure prints the results after a record has been processed }
{ onto the printer, in the format of the form }

Procedure PrintRecord2;
Var Total1, D1 : integer;
Begin
  clrscr;
If degree='BFA' then total1:=34 else if
degree='BM' then total1:=37 else if
degree='BME' then total1:=39;

If degree='BFA' then D1:=4 else if
degree='BM' then D1:=6 else if
degree='BME' then D1:=4;

writeln(lst,'ILLINOIS WESLEYAN UNIVERSITY':53);
writeln(lst,' ':28,Degree,' CREDIT CHECK FORM ');
write(lst,' Student Name : ',Name);
For i:=length(Name) to 25 do
  write(lst,' ');
write(lst,'Advisor Name : ',Advisor);
write(lst,' Major1 : ',Major);
For i:=length(Major) to 25 do
  write(lst,' ');
write(lst,'Major2 ; ',Major2);
writeln(lst);writeln(lst);
write(lst,'GENERAL UNIVERSITY REQUIREMENTS':54);
write(lst);writeln(lst);
If Eng.Depts=" then write(lst,' ');
else
  write(lst,' O.K. ');
write(lst,' I. writing: English 105 or equivalent: ');
If Eng.Depts<>" then write(lst,Eng.Depts,' ',Eng.Numbers)
else write(lst,' ');
writeln(lst);writeln(lst);
If Flang2.Depts=" then write(lst,' ');
else
  write(lst,' O.K. ');
write(lst,' II. Foreign Language: ');
If Flang.Depts<>" then write(lst,Flang.Depts,' ',Flang.Numbers)
else write(lst,' ');
If Flang2.Depts<>" then write(lst,' ',Flang2.Depts,' ',Flang2.Numbers)
else write(lst,' ');
If (degree='BME') then
If Flang3.Depts<>" then write(lst,' ',Flang2.Depts,' ',Flang2.Numbers)
else write(lst,' ');
writeln(lst);writeln(lst);
If Rel.Depts=" then write(lst,' ');
else
  write(lst,' O.K. ');
write(lst,' III. Religion: ');
If Rel.Depts<>" then write(lst,Rel.Depts,' ',Rel.Numbers)
else write(lst,' ');
writeln(lst);writeln(lst);
write(lst,'IV. Humanities: ');
If Hcount>1 then
  writeln(lst,' O.K. ',' IV. Humanities: ')
else writeln(lst,' ',' IV. Humanities: ');
writeln(lst);
writeln(lst,'Two courses from at least two of the following':61);
writeln(lst,'areas:':21);
writeln(lst);
write(lst,' 1. Literature ');
If Lit1.Depts<>'' then writeln(lst,' ',Lit1.Depts,' ',Lit1.Numbers)
else writeln(lst,' ');
write(lst,' 2. Philosophy ');
If Phil1.Depts<>'' then writeln(lst,' ',Phil1.Depts,' ',Phil1.Numbers)
else writeln(lst,' ');
write(lst,' 3. Humanities ');
If Hum1.Depts<>'' then writeln(lst,' ',Hum1.Depts,' ',Hum1.Numbers)
else writeln(lst,' ');
writeln(lst);
write(1st, ', 2. Philosophy ');
If Phil1.Depts<>'' then writeln(lst,' ',Phil1.Depts,' ',Phil1.Numbers)
else writeln(lst,' ');
writeln(lst); If SScount=3 then
write(lst,' O.K. V. Social Science :')
else writeln(lst,' ');
writeln(lst);writeln(lst);
write(lst,' 3. Humanities ');
If Hum1.Depts<>'' then writeln(lst,' ',Hum1.Depts,' ',Hum1.Numbers)
else writeln(lst,' ');
writeln(lst);writeln(lst);writeln(lst);writeln(lst);
write(lst,' Economics: ');
If Econ.Depts<>'' then write(lst,' ',Econ.Depts,' ',Econ.Numbers)
else write(lst,' ');
writeln(lst);
write(lst,' History: ');
If Hist.Depts<>'' then write(lst,' ',Hist.Depts,' ',Hist.Numbers)
else write(lst,' ');
writeln(lst);
write(lst,' Political Science: ');
If Psci.Depts<>'' then write(lst,' ',Psci.Depts,' ',Psci.Numbers)
else write(lst,' ');
writeln(lst);
write(lst,' Sociology (not 291): ');
If Soc.Depts<>'' then write(lst,' ',Soc.Depts,' ',Soc.Numbers)
else write(lst,' ');
writeln(lst);
write(lst,' Social Science: ');
If sosc.Depts<>'' then write(lst,' ',sosc.Depts,' ',sosc.Numbers)
else write(lst,' ');
writeln(lst); writeln(lst);
If ((Degree='BM') or (Degree='BFA')) then
Begin
If (NScount>1) then
Begin
writeln(lst,' O.K. VI. Natural Science: at least two course units from
writeln(lst,' two of the following groups')
End
else
Begin
writeln(lst,'VI. Natural Science: at least two course units from
writeln(lst,' at least two of the followin
End;
writeln(lst);
write(lst,' Group 1: (biology, psychology, NatSci 101) ');
If Life.Depts<>'' then writeln(lst,Life.Depts,' ',Life.Numbers)
else writeln(lst,' ');
write(lst,' Group 2: (chemistry, physics) ');
If PhSc.Depts<>'' then writeln(lst,PhSc.Depts,' ',PhSc.Numbers)
else writeln(lst,' ');}
write(lst,' Group 3: (mathematics, computer science) ') ;
If MaCs.Depts<>" then writeln(lst,MaCs.Depts,' ',MaCs.Numbers)
else writeln(lst,'________');
writeln(lst);
End;

If (Degree='BME') then

Begin
If (NScount1>2) and (Labsc.Depts<>"')) then
  writeln(lst,' O.K. VI. Natural Science: at least 3 course units from (b
else
  writeln(lst,'________ VI. Natural Science: at least 3 course units from
  writeln(lst,'________ chemistry, or physics) and at least one labora
  writeln(lst,'________ science: ');
write(lst,' ',Labsc.Depts,' ',Labsc.Numbe
else write(lst,' ');
If NS1.Depts<>" then write(lst,' ',NS1.Depts,' ',NS1.Numbers)
else write(lst,'________ ');
If NS2.Depts<>" then write(lst,' ',NS2.Depts,' ',NS2.Numbers)
else write(lst,'________ ');
If NS3.Depts<>" then write(lst,' ',NS3.Depts,' ',NS3.Numbers)
else write(lst,'________ '); writeln(lst);
End;
If Pecount>3 then
  writeln(lst,' O.K. VII. Physical Education :')
else
  writeln(lst,'________ VII. Physical Education :');
write(lst,'________ 2 courses (X) or 4 half courses (Y) or an equiva
write(lst,'________ combination: ');
If Pec1.Depts<>" then write(lst,' ',Pec1.Depts,' ',Pec1.Numbers)
else write(lst,'________ ');
If Pec2.Depts<>" then write(lst,' ',Pec2.Depts,' ',Pec2.Numbers)
else write(lst,'________ ');
If Pec3.Depts<>" then write(lst,' ',Pec2.Depts,' ',Pec3.Numbers)
else write(lst,'________ ');
If Pec4.Depts<>" then write(lst,' ',Pec4.Depts,' ',Pec4.Numbers)
else writeln(lst,'________ '); writeln(lst);
If degree='BM' then

Begin
  If Elec.Depts='' then write(lst,'________ ')
else
  write(lst,' O.K. ');
  write(lst,'________ VIII. Elective: 1 course unit non-music elective ');
If Elec.Depts<>'' then write(lst,Elec.Depts,' ',Elec.Numbers)
else write(lst,'________ ');
End;

If degree='BME' then

Begin
  If Math.Depts='' then write(lst,'________ ')
else

write(lst,' O.K. ');
write(lst,' VIII. Mathematics: one course unit ');
   else write(lst,'------');
End;

writeln(lst); writeln(lst);
write(lst,Higher:0:2);
If Higher<10 then write(lst,' ');
writeln(lst,' Total completed (11 course units numbered 300 and 400 are ':61
   writeln(lst, '' required with at least 4 in a departmental major)'':74);
writeln(lst);
write(lst,Tunit:0:2);
If Tunit>9.99 then write(lst,',' ')
   else write(lst,',' ');
write(lst,' Total units completed ('',Total1,' course units required')');
writeln(lst); writeln(lst);
write(lst,Dunit:0:2,' Total "D" units (no more than ',D1,' will be counted ')
   writeln(lst);
write(lst,Dmajor:0:2,' Total "D" units in major1 : ',major);
write(lst,' (no more than 1 will be counted ');
write(lst,Dmajor2:0:2,' Total "D" units in major2 : ',major2);
write(lst,' (no more than 1 will be counted ');
write(lst,' toward graduation');
writeln(lst);
writeln(lst);
write(lst,' Student Signature: ________________ Date: ____________');
writeln(lst);
writeln(lst,' Advisor Signature: ________________ Date: ____________');
End;

*************************************************************************
{ Only for the BSN degree}
{ This procedure prints the results after a record has been processed }
{ onto the screen, in the format of the form }
else write('______');
writeln; writeln;

If Fart.Depts='' then write('______')
else
  write('  O.K. ');
write('   II. Fine Arts: ');
If Fart.Depts<'' then write(Fart.Depts,' ',Fart.Numbers)
else write('______');
writeln; writeln;

If ((Log.Depts<'' and (Rel.Depts<'' and (PhilIN.Depts<''
  and (HumLit.Depts<'')) then
  writeln('  O.K. ',' III. Humanities: ')
else writeln(' ', III. Humanities: ');
writeln;
write(' A. Logic: ');
If Log.Depts<'' then writeln(' ',Log.Depts,' ',Log.Numbers)
else writeln('______ ');
write(' B. Religion: ');
If Rel.Depts<'' then writeln(' ',Rel.Depts,' ',Rel.Numbers)
else writeln('______ ');
writeln('  C. Philosophy: One course in philosophy chosen from');
writeln('  103, 104, 109, 110, 111, 120, 270, 271, or an');
write('appropriate 112: ');
If PhilIN.Depts<'' then writeln(' ',PhiIN.Depts,' ',PhiIN.Numbers)
else writeln('______ ');
write(' D. Humanities, Literature: one course: ');
If Humlit.Depts<'' then writeln(' ',Humlit.Depts,' ',Humlit.Numbers)
else writeln('______ ');
delay(3000); clrscr;
writeln;
If SScount>2) and (SS102.Depts<'' then
  writeln(' O.K. IV. Social Science: ')
else
  writeln(' IV. Social Science: Three course units to include:');
writeln; writeln;
write(' A. Social Science 102: ');
If SS102.Depts<'' then write(' ',SS102.Depts,' ',SS102.Numbers)
else write('______ ');
writeln; writeln;
write(' B. Two course units chosen from:');
write('Economics: ');
If Econ.Depts<'' then write(' ',Econ.Depts,' ',Econ.Numbers)
else write('______ ');
writeln;
write('History: ');
If Hist.Depts<'' then write(' ',Hist.Depts,' ',Hist.Numbers)
else write('______ ');
writeln;
write('Political Science: ');
If Psci.Depts<'' then write(' ',Psci.Depts,' ',Psci.Numbers)
else write('______ ');
writeln;
write('Sociology (not 291): ');
If Soc.Depts<'' then write(' ',Soc.Depts,' ',Soc.Numbers)
• else write(' '); writeln;
write(' ', A. Human Biology 107 and 108 ');
else write(' '); writeln;
else write(' '); writeln;
write(' ', B. Chemistry 110: ');
else write(' '); writeln;
write(' ', C. Natural Science: ');
If NS101.Depts<>'' then write(' ',NS101.Depts,' ',NS101.Numbers)
else write(' '); writeln;

delay (3000); clrscc; writeln; writeln;
if MaCS.Depts=' then write(' ')
else write( , O. K. ');
write( VI. Mathematics or Computer Science: ');
If MaCS.Depts<>'' then write(MaCS.Depts,' ',MaCS.Numbers)
else write(' '); writeln;

If Pecount>3 then
writeln(' O.K. VII. Physical Education :')
else writeln(' ______ VII. Physical Education :');
writeln;

writeln(' 2 courses (X) or 4 half courses (Y) or an equivalent:');
write(' combination: ');
If Pec1.Depts<>'' then write(' ',Pec1.Depts,' ',Pec1.Numbers)
else write(' '); writeln;
If Pec2.Depts<>'' then write(' ',Pec2.Depts,' ',Pec2.Numbers)
else write(' '); writeln;
If Pec3.Depts<>'' then write(' ',Pec3.Depts,' ',Pec3.Numbers)
else write(' '); writeln;
else writeln(' ______
writeln; writeln; writeln;
write(Higher: 0:2); If Higher<10 then write(' ');}
writeln('Total completed (11 course units numbered 300 and 400 are required with at least 4 in a departmental major)');
writeln;
write(Tunit:0:2);
If Tunit>9.99 then write('')
else write('');
write('Total units completed (35 course units required)');
writeln; writeln;
write(Dunit:0:2,' Total "D" units (no more than 4 will be counted toward g');
writeln;
write(Dmajor:0:2,' Total "D" units in major1 : ',major);
writeln(' (no more than 1 will be counted toward graduation)');
writeln;
while not(keypressed) do;
readln;
End;

{***********************************************************************}
{ Only for the BSN degree }
{ This procedure prints the results after a record has been processed }
{ onto the printer, in the format of the form }

Procedure PrintRecord3;

Var Total1, D1 : integer;

Begin
clrscr;
writeln(lst,'ILLINOIS WESLEYAN UNIVERSITY':53);
writeln(lst, '28, Degree, CREDIT CHECK FORM ');
writeln(lst);
writeln(lst);
write(lst,'Student Name: ',Name);
For i:=length(Name) to 25 do
write(lst,' ');
writeln(lst,'Advisor Name: ',Advisor);
writeln(lst); writeln(lst);
writeln(lst,'GENERAL UNIVERSITY REQUIREMENTS':54);
writeln(lst);
If Eng.Depts='' then write(lst,'_______')
elserwrite(lst,' O.K. ');
write(lst,' I. Writing: English 105 or equivalent: ');
If Eng.Depts<>'' then write(lst,Eng.Depts,' ',Eng.Numbers)
elsewrite(lst,'_______');
writeln(lst); writeln(lst);
If Part.Depts='' then write(lst,'_______')
elserwrite(lst,' O.K. ');
write(lst,' II. Fine Arts: ');
If Part.Depts<>'' then write(lst,Part.Depts,' ',Part.Numbers)
elsewrite(lst,'_______');
writeln(lst); writeln(lst);
If ((Log.Depts<>'' and (Rel.Depts<>'' and (PhilN.Depts<>''))
 and (HumLit.Depts<>''))) then
writeln(lst,' 0.K. ', III. Humanities: '
else writeln(lst,'_______', III. Humanities: ');}
writeln(lst);
write(lst,' A. Logic: ');
If Log.Depts<>' then writeln(lst,' ',Log.Depts,' ',Log.Numbers)
  else writeln(lst,' ------ ');
write(lst,' B. Religion: ');
If Rel.Depts<>' then writeln(lst,' ',Rel.Depts,' ',Rel.Numbers)
  else writeln(lst,' '));
writeln(lst,' C. Philosophy: One course in philosophy chosen from
write(lst,' 103, 104, 109, 110, 111, 120, 270, 271, or an ');
write(lst,' appropriate 112: ') );
If PhiIN.Depts<>' then writeln(lst,' ',PhiIN.Depts,' ',PhiIN.Numbers)
  else writeln(lst,' ------ ');
write(lst,' D. Humanities, Literature: one course: ');
If Humlit.Depts<>' then writeln(lst,' ',Humlit.Depts,' ',Humlit.Numbers)
  else writeln(lst,' ------ ');
writeln(lst);
If ((SScount>2) and (SS102.Depts<>')) then writeln(lst,' O.K. IV. Social Science : ')
  else writeln(lst,' ------ IV. Social Science: Three course units to includ
  write(lst,' A. Social Science 102: ');
If SS102.Depts<>' then write(lst,' ',SS102.Depts,' ',SS102.Numbers)
  else write(lst,' ------ ');
writeln(lst);
writeln(lst,' B. Two course units chosen from: ');
write(lst,' E. Economics: ');
If Econ.Depts<>' then write(lst,' ',Econ.Depts,' ',Econ.Numbers)
  else write(lst,' ------ ');
writeln(lst);
write(lst,' History: ');  
If Hist.Depts<>' then write(lst,' ',Hist.Depts,' ',Hist.Numbers)
  else write(lst,' ------ ');
writeln(lst);
write(lst,' C. Political Science: ');
If Psci.Depts<>' then write(lst,' ',Psci.Depts,' ',Psci.Numbers)
  else write(lst,' ------ ');
writeln(lst);
write(lst,' Sociology (not 291): ');
If Soc.Depts<>' then write(lst,' ',Soc.Depts,' ',Soc.Numbers)
  else write(lst,' ------ ');
writeln(lst);
writeln(lst);
If ((Bio107.Depts<>' and (Bio108.Depts<>'
and (Chem110.Depts<>' and (NS101.Depts<>')) ) then writeln(lst,' O.K. V. Natural Science: ')
  else writeln(lst,' ------ V. Natural Science: ');
writeln(lst);
write(lst,' A. Human Biology 107 and 108 ');  
  else write(lst,' ------ ');
If Bio108.Depts<>' then write(lst,' ',Bio108.Depts,' ',Bio108.Numbers)
  else write(lst,' ------ ');
writeln(lst);
write(lst,' B. Chemistry 110: ');
If Chem110.Depts<>'' then write(lst,' ',Chem110.Depts,' ',Chem110.Numbers)
else write(lst,'______');
writeln(lst);
write(lst,' C. Natural Science: ');
If NS101.Depts<>'' then write(lst,' ',NS101.Depts,' ',NS101.Numbers)
else write(lst,' ');
writeln(lst) ; writeln(lst);
write(lst, ' VI. Mathematics or Computer Science: ');
If MaCS.Depts<>'' then write(lst, MaCS.Depts,' ',MaCS.Numbers)
else write(lst,' O.K. ');
write(lst,' VII. Physical Education :')
else writeln(lst,'______ VII. Physical Education :');
writeln(lst);
write(lst,' O.K. VII. Physical Education :')
else writeln(lst,'______ VII. Physical Education :');
writeln(lst);
write(lst,' Total completed (11 course units numbered 300 and 400 are ');
write(lst,':61 required with at least 4 in a departmental major)':74);
writeln(lst);
write(lst,' Total units completed (35 course units required)');
writeln(lst); writeln(lst);
write(lst,' Total "D" units (no more than 4 will be counted toward');
writeln(lst);
write(lst,' Total "D" units in major1: ',major);
writeln(lst,' (no more than 1 will be counted ');
writeln(lst,' toward graduation');
writeln(lst);
writeln(lst,' Student Signature: __________________ Date: ______________
writeln(lst);
writeln(lst,' Advisor Signature: __________________ Date: ______________
End;
{*************************************************************************}
Begin {Main}
initialize;  { Initializes all the variables }
assign(Datafile,'c:\master.dat');
reset(Datafile);
readdata;
choice:='';
Copyright;
readln;

{ This loop helps execute the choices from the main menu }

While ((Choice<>q') and (Choice<>Q')) do

Begin {While loop}
clrscr;
printmenu(Choice);

if choice='1' then  { 1. Add new student }
Begin
adddata(counter);
Temp:=Datarray[Counter];
eight:=eight+'. '+Temp[9];
if new in ['Y','Y'] then
Begin
initialize;
assign(Infofile,Temp);
addinfo;
End;
End;

if choice='2' then  { 2. Delete a record }
Begin
clrscr;
check:=false;
count:=1;
write('Input social security # of the record you want to delete : '); readln(Temp);
eight:=temp;
temp:=temp+'.'+temp[9];
while ((temp<>Datarray[Count]) and (Count<Counter)) do
Begin
   count:=count+1;
End;

if temp=Datarray[count] then
check:=true;
write('Are you sure (Y/N): '); readln(Sure);
If (check=true) and (sure in ['y','Y']) then
Begin
assign(Infofile,Temp);
datarray[count]:='';
write('Record deleted');
write('Record deleted');
write('Record deleted');
write('Record deleted');
write('Record deleted');
Temp:=Eight+Temp[10];
writeln('Record ','Temp',' has been deleted');
write(chr(7),chr(7),chr(7));
delay(2000);
erase(Infofile);
End
else
Begin
writeln; writeln;
If (check=false) then
Begin
writeln('Record not found !!!!!!!!!':52);
writeln; writeln;
End;
If ((sure in ['n','N']) and (check=true))then
Begin
Temp:=Eight+Temp[10];
writeln('Record ','Temp',' has not been deleted');
End;
write(chr(7),chr(7),chr(7));
delay(2000);
write(chr(7),chr(7),chr(7));
Gotoxy(1,18);
write('Press Any Key To Continue..........':77);
while not(keypressed) do;
readln;
End;
End; {End Choice =2}

If choice='3' then { 3. Edit an existing record }
Begin
Initialize;
GetRecord;
If check=true then
Begin
Request1:=true;
Request2:=true;
assign(Infofile,Num);
ReadRecord;
EditRecord;
Writerecord;
End;
End; {End Choice=3}

If choice='4' then { 4. Process and view a record }
Begin
Initialize;
GetRecord;
If check=true then
Begin
assign(Infofile,Num);
ReadRecord;
If ((degree='BFA') or (degree='BME') or (degree='BM')) then
Begin
ProcessRecord;
ProcessHum;
ProcessNS;
ProcessNS2;
If degree='BM' then
ProcessElec;
WRecord2;
End;
If (degree='BSN') then
Begin
ProcessRecord;
ProcessFA;
ProcessOthers;
ProcessNS;
ProcessPE;
ProcessSS;
ProcessBSN;
WRecord3;
End;
End; {End Choice=4}

If choice='5' then { Process a print a record onto the printer }
Begin
Initialize;
GetRecord;
If check=true
then
Begin
assign(Infofile,Num);
ReadRecord;
if ((degree='BFA') or (degree='BM') or (degree='BME'))then
Begin
ProcessRecord;
ProcessHum;
ProcessNS;
ProcessNS2;
ProcessSS;
ProcessPE;
ProcessOthers;
ProcessOthers2;
If degree='BM' then
ProcessElec;
PrintRecord2;
End
else if (degree='BSN') then
Begin
ProcessRecord;
ProcessFA;
ProcessOthers;
ProcessNS;
ProcessPE;
ProcessSS;
ProcessBSN;
PrintRecord3;
End;
End; {End Choice=5}
End; {While Loop}
writedata(Counter);
close(Datafile);
End. (Main)