Information and Communication Technology (Coursework)

Option D : Software Development

Title : Seating plan

Contents

1.Introduction

1.1 Background

ABC school is going to organize a dinner for alumni to celebrate its 50th Anniversary. The school will develop a program for the dinner registration and generating a seating plan.

1.2 Requirements

After the data collection stage, the personal information of the participants will be input into the program, as shown in the following example:

- name of participant
- year of graduation
- sex
- age
- employment
- number of seats required

Then, I would design three methods which allow user to choose two of them to sort

the participants. Which are sort by:

- Graduation year
- Balancing gender
- Name order

Finally, it generates a UI page to fill with the sorted participants.

2.Design

2.1 Description

Base on the situation and information provided in Chapter 1. I will design the following things:

1. Login and register system & UI page
2. Menu for basic management of seating plan
   i. Option for viewing and editing
3. Functions of program
   i. Add and save participants into a text file
   ii. Allow insertion or deleting participants
   iii. Sorting

Assume that :

1. The maximum capacity of the venue is 64
2. Only have to separate into two tables
3. Only one person is authorized to manage this plan

2.2 Features

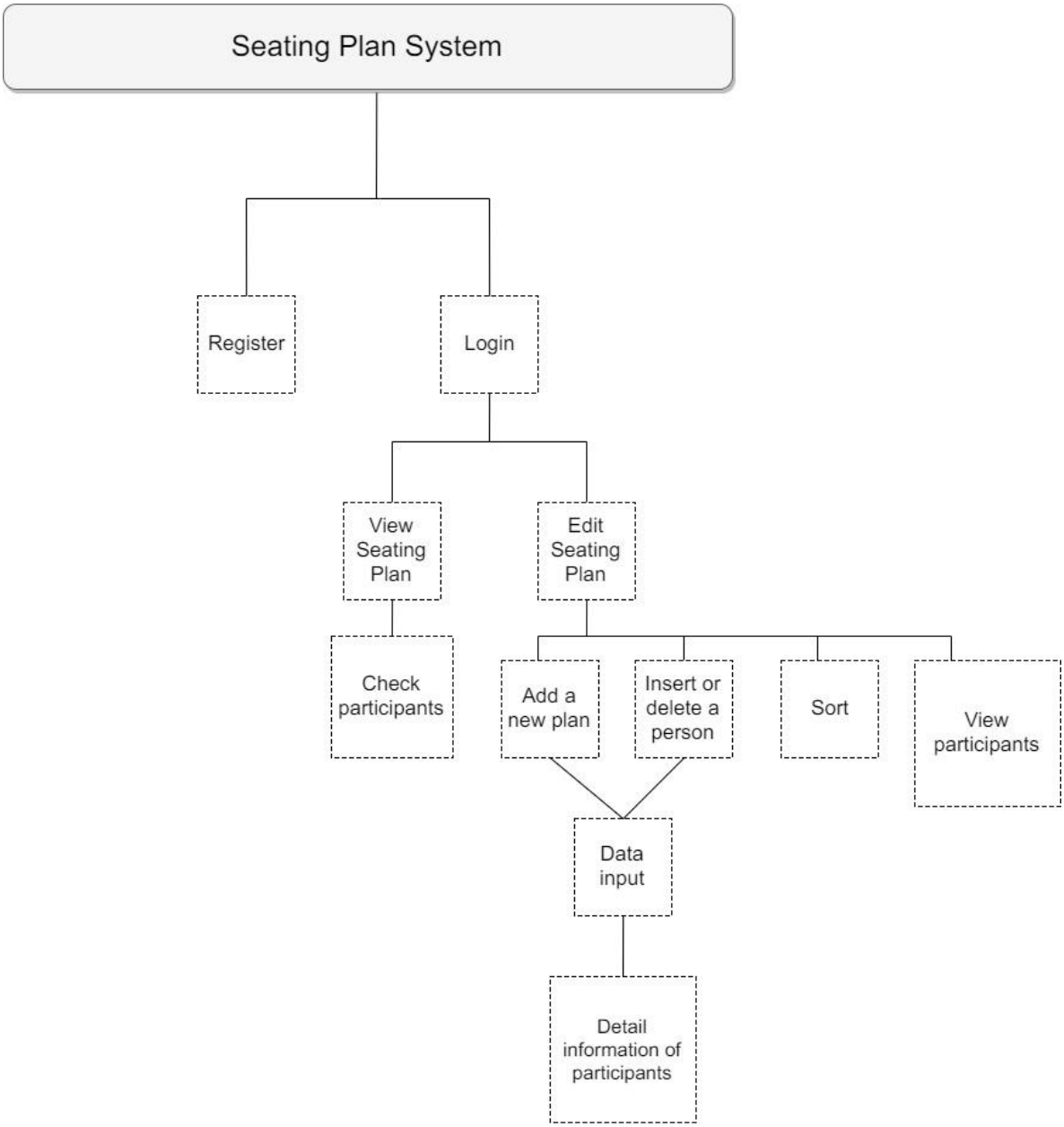
The expected user is the person in charge of allocating seat in the dinner. And the person is able to input the data of the participants, choose two desire sorting method, and view the seating plan according to the allocated seating number efficiently. Modification is allowed like insert and delete (a) person. The program will be designed to be user-friendly, UI and clear instruction or indicator is a must. So that the person can know how to use it quickly.

The features are as following :

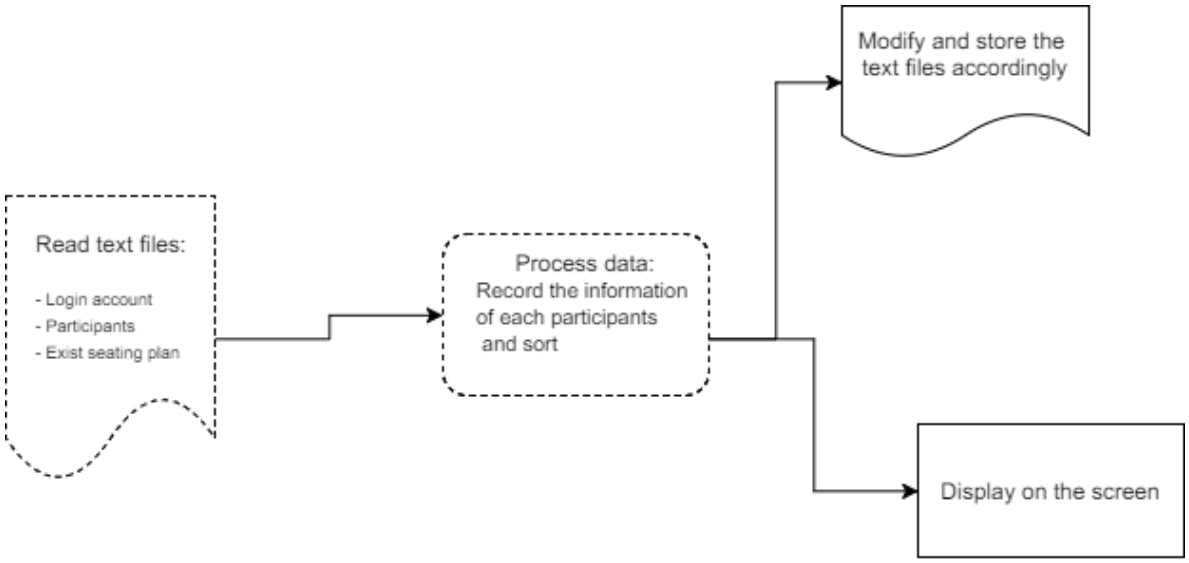
Features	**Function**
Register & Login	Allow user to create and login to their account
Add a new seating plan	Delete everything stored in text file and create a new one
Insert a participant	Allow user to insert a participant into the current seating plan
Delete a participant	Allow user to delete a participant of the current seating plan by input the name
Sorting	For user to choose two desire method to sort the participants for the current seating plan
View participants	For user to check the input data is correct by showing all the participates' name, age and gender

View seating plan	For user to inspect the sorted seating plan also allow to check the detail information of the participants

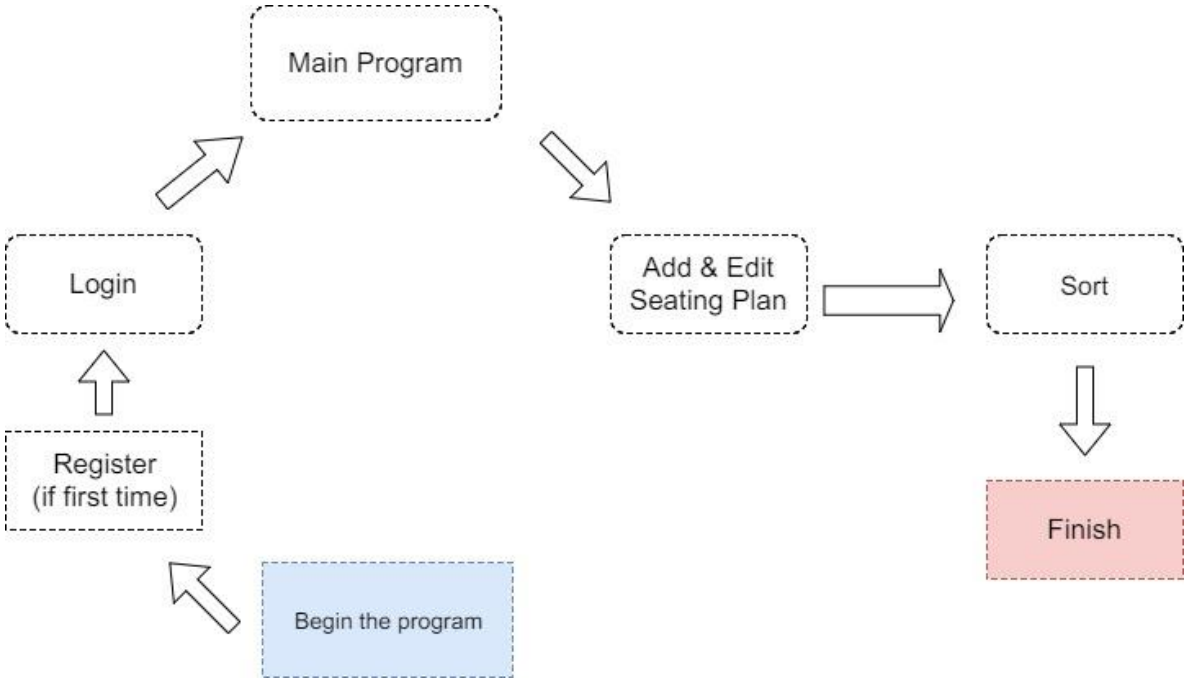
Main program



Data flow



Processing



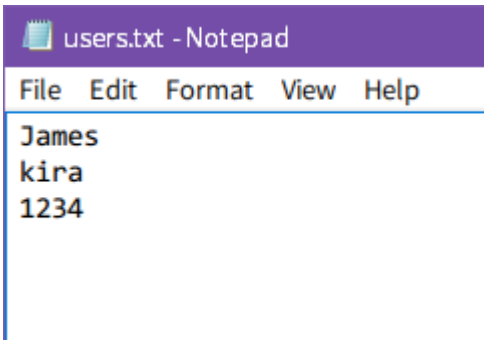
2.3 Database

In order to store the user's account, participants' detail and the data after sorted, there will be 3 database files assigned. The complexity and the smooth of program is highly dependent to these files. Since the data amount is huge, using database can help improve the accuracy of reading data, more precisely, reliable and enhance the efficiency of the whole program. These text files are as following :

Data1: users.txt

This file stores the user data, name, login id and password will be recorded. It's not quite necessary to have a login system build-in, especially for one user only. But I think it can still prevent the unauthorized action. Once the register action is done, the file will be empty first and record the new data. This file will be read only once or after register.

Structure:



1st line : User name (Strings

2nd line : login id (Strings

3rd line : password (Strings

Data2: participants.txt

This file stores all the participants' information. It updates only when the following actions were taken(Add, insert, delete). Also, this file will be always the first priority to be read due to the procedure purpose. To make sure that every participants' data are read, and every change made are valid and correspondingly.

Structure:

participants.txt - Notepad

File Edit Format View Help

```
Teri
27
F
2005
Employee
Jean
27
M
2006
Employee
Marina
29
F
2013
Employee
Shane
25
F
2009
Employee
Gary
26
M
2008
Employee
Lorayne
23
F
2005
Employee
Mary
24
F
2010
Employee
Jena
```

1st line : Name of the participant (Strings

2nd line : Age (Integers

3rd line : Gender (Character

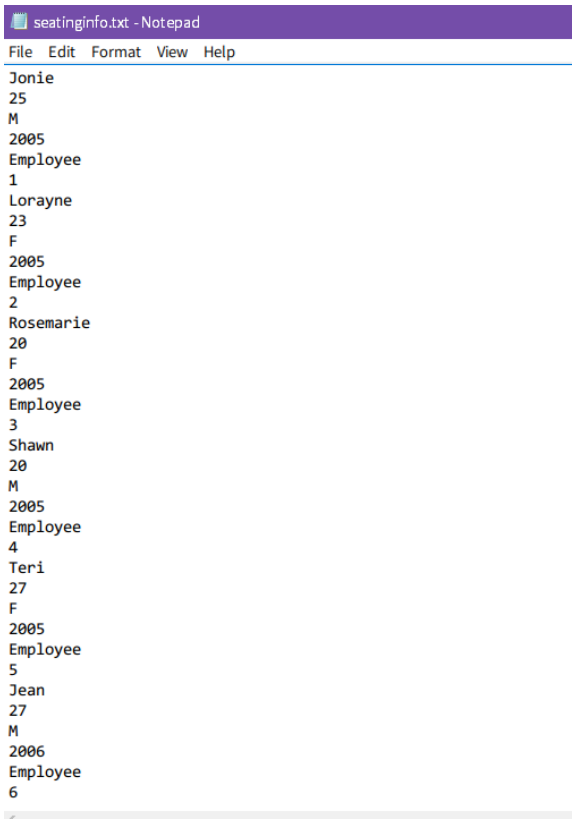
4th line : Graduate year (Integers

5th line : Employment (Strings

Data3: seatinginfo.txt

This file stores the sorted information of all participants, and assign a seating number respectively. It updates only after the sort action has been done, these data are based in Data2, prevent overlapping among two files when processing data. So, this is also a final outcome file.

Structure:



```
Jonie
25
M
2005
Employee
1
Lorayne
23
F
2005
Employee
2
Rosemarie
20
F
2005
Employee
3
Shawn
20
M
2005
Employee
4
Teri
27
F
2005
Employee
5
Jean
27
M
2006
Employee
6
```

1st line : Name of the participant (Strings

2nd line : Age (Integers

3rd line : Gender (Character

4th line : Employment (Strings

5th line : Seating number (Integers

3.Implementation

3.1 Data Structure

Library used: Crt

By using this library, it supports some other external functions such as cleaning the current screen, allows changing the color of text. These functions are being used frequently, and providing better experience for the user since a better instruction can be shown on the screen.

Global variables:

Arrays:

These arrays store all the participants' information, and responsible for the record of database. Being read and update frequently due to the situation.

```
{       name     : array[1..64] of string;
        age      : array[1..64] of integer;
        sex      : array[1..64] of char;
        grad     : array[1..64] of integer;
        job      : array[1..64] of string;
        seat_num : array[1..64] of integer;     }
```

Strings:

These strings store the data required for login system. Being read and update only in the function of register and login system.

```
{       username,userid,userpw : string;       }
```

Integers:

These integers store the data that being used frequently, especially for different procedure to make use of those.

```
{       num_ppl, user_d, stp, npp : integer;   }
```

Booleans:

These Booleans responsible for better indicating and tell the program to display which screen after different actions are done. Such as you don't want the program to show the login screen after you have logon and just finished the sort action. So, it can direct the program to proper screen due to the user's action were taken.

```
{       logon, inmenu, insort, inedit, exx : boolean;  }
```

3.2 Procedures

Editing participants' information:

For the first time to use this program and input all the participants' detail information, or just want to reset all the participants and input again. Then all the data will be save to text file immediately.

```
procedure add_ppl;
var
  a, nump, age1, grad1 : integer;
  name1, job1 : string;
  sex1 : char;

begin
 Clrscr;
 inmenu := false;
 a := 0;
 num_ppl := 0;
 nump := 0;
 write('Current participants :');
 textcolor(yellow);
 write(a);
 textcolor(white);
 write(' ');
 write('(Maximum 64)');
 writeln;
 writeln;
 writeln;
 write('Please enter the number of participants : ');
 readln(nump);
 for a := 1 to nump do
  begin
  Clrscr;
  write('Current participants :');
  textcolor(yellow);
  write(a - 1);
  textcolor(white);
  write(' ');
  write('/', nump);
  writeln;
  writeln;
  writeln;
  write('Please enter the name : ');
  readln(name1);
  writeln;
  write('Please enter the age of ', name1, ': ');
  readln(age1);
  writeln;
  write('Please enter the sex of ', name1, ': ');
  readln(sex1);
  writeln;
  write('Please enter the year graduated of ', name1, ': ');
  readln(grad1);
  writeln;
  write('Please enter the employment of ', name1, ': ');
  readln(job1);
  writeln;
      name[a] := name1;
      age[a] := age1;
      sex[a] := sex1;
      grad[a] := grad1;
      job[a] := job1;
      num_ppl := num_ppl + 1;
      storepp;
  end;
end;
```

Then, if the user wishes to modify the input data, two functions will be provided for insert or delete participant(s).

Procedure for insert:

```
procedure insert_ppl;
var
  k,age1,grad1 : integer;
  name1, job1 : string;
  sex1 : char;
  f : text;

begin
  Clrscr;
  inmenu := false;
  assign(f,'participants.txt');
  if stp = 64 then
  begin
  textcolor(red);
  writeln;
  writeln;
  writeln('                All tables full !          ');
  writeln('        You cannot add person anymore!!  ');
  textcolor(white);
  readln;
  end
  else
  begin
    read_info;
    Clrscr;
    k:=stp+1;
    writeln(k);
    write('Current participants :');
    textcolor(yellow);
    write(stp);
    textcolor(white);
    write(' ');
    write('/64');
    writeln;
    writeln;
    writeln;
    write('Please enter the name : ');
    readln(name1);
    writeln;
    write('Please enter the age of ', name1, ': ');
    readln(age1);
    writeln;
    write('Please enter the sex of ', name1, ': ');
    readln(sex1);
    writeln;
    write('Please enter the year graduated of ', name1, ': ');
    readln(grad1);
    writeln;
    write('Please enter the employment of ', name1, ': ');
    readln(job1);
    writeln;
        append(f);
        name[k] := name1;
        age[k] := age1;
        sex[k] := sex1;
        grad[k] := grad1;
        job[k] := job1;
        num_ppl := num_ppl + 1;
        close(f);
        store_newpp;
    readln;
  end
end;
```

Procedure for delete:

```
procedure del_ppl;
var
  i, j : integer;
  FindName : string;
begin
  Clrscr;
  inmenu := false;
  write('Please enter the name of the person: ');
  readln(FindName);
  j := 0;
  for i := 1 to stp do
    if name[i] = FindName
      then j := i;
  if j > 0 then
    begin
      for i := j to stp -1 do
      begin
      name[i] := name[i+1];
      age[i] := age[i+1];
      sex[i] := sex[i+1];
      grad[i] := grad[i+1];
      job[i] := job[i+1];
     end;
     store_mpp;
     writeln('This person has been deleted.');
     readln;
     stp := stp - 1
   end
  else
  begin
  textcolor(red);
  writeln('No such person!');
  textcolor(white);
  readln;
  end;
end;
```

Also, there has a function for user to view the input data and able to search by name or view all the participants' data. So, it provides a method to valid the input data. If the user chooses to search by name, the procedure is as follow:

```
procedure searchppl;
var
  i : integer;
  op : char;
  nametp : string;
  found : boolean;

begin
 writeln;
 write  ('              Please enter the name of the participant you want to search : ');
 readln(nametp);
 found := false;
 i := 1;
 while (i < stp +1) and (not found) do
   begin
   if nametp = name[i] then
     begin
     found := true;
     end
     else
     begin
     found := false;
     i := i +1;
     end;
   end;
   if (found = true) then
    begin
    writeln('              You are searching : ',name[i]);
    writeln('              Gender : ', sex[i]);
    writeln('              Age : ', age[i]);
    writeln('              Graduated in : ', grad[i]);
    writeln('              Employment : ', job[i]);
    writeln();
    end
    else
    begin
    textcolor(red);
    writeln('              No such person, please check again ');
    textcolor(white);
    writeln();
    end;
 write('              Continute to search ? (Y/N) : ');
 readln(op);
 if (op = 'N') then
    begin
    inmenu := false;
    inedit := true;
    end
 else
    begin
    searchppl;
    end;


end;
```

If the user chooses to view all the participants, the procedure will first sort all of them by name order before displaying on the screen, the procedure is as follow:

```
procedure sortplan_name;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if name[i] > name[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;

procedure displayppl;
var i : integer;
begin
    Clrscr;
    inmenu := false;
    sortplan_name;
    writeln('          List of participants:         ',' (',stp,') ');
    writeln;
    writeln('          Name      Age   Gender');
    writeln('================================');
    textcolor(yellow);
    for i:= 1 to stp do
      begin
      writeln('  ',i:2,' : ',name[i]:9, age[i]:5, sex[i]:5);
      end;
    textcolor(white);
    readln;
end;
```

Once it’s confirmed that all the data are inputted correctly, user can now choose the combination of sorting method. The program will use bubble sort and there are three sorting way provided, which are graduation year, balancing gender and name order. It’s needed to add conditional code due to the combination of sorting. For example, once I choose to sort by graduation year together with name order. The procedure will first sort all the participants in ascending order of graduation year, then an extra condition when sort by name order is required. Otherwise, it’s meaningless because the name order would affect the whole order of the previous sorting. So similar procedure with different condition will be shown, and I will mark those different line in code.

First one is combination of graduation year and balancing gender:

```
procedure gradgendercombo;
    begin
    sortplan_grad;
    sortplan_1grad2gender;
    assign_seat;
    storeseat;
writeln('==========================');
    textcolor(green);
writeln('    Success !       ');
textcolor(white);
writeln('==========================');
    readln;
    end;

procedure sortplan_grad;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if grad[i] > grad[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;
```

```
procedure sortplan_1grad2gender;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if grad[i] = grad[i+1] then
      if sex[i] <> sex[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;
```

The above conditions can make sure that the function take place only when the graduation year is same to the next one with different gender.

The second combination is follow the graduation year with name in ascending order:

```
procedure gradnamecombo;
begin
    sortplan_grad;
    sortplan_1grad2name;
    assign_seat;
    storeseat;
writeln('==========================');
    textcolor(green);
writeln('    Success !       ');
textcolor(white);
writeln('==========================');
    readln;
end;

procedure sortplan_grad;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;
begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if grad[i] > grad[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;
```

```
procedure sortplan_1grad2name;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if grad[i] = grad[i+1] then
      if name[i] > name[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;
```

The above conditions can make sure that the function take place only when the graduation year is same to the next one, and base on this situation, sort by name in ascending order.

The third combination is follows the name order with balancing gender:

```
procedure namegendercombo;
begin
    sortplan_name;
    sortplan_1name2gender;
    assign_seat;
    storeseat;
writeln('==========================');
    textcolor(green);
writeln('    Success !       ');
textcolor(white);
writeln('==========================');
    readln;
end;


procedure sortplan_name;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;
begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if name[i] > name[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;
```

```
procedure sortplan_1name2gender;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if name[i] > name[i+1] then
      if sex[i] <> sex[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;
```

The above conditions can make sure that the function take place only when the first alphabet of the name is larger than that of the next one. And then balance the gender.

Once the sorting action been done. The relative data file will be updated immediately and pair them by seating number. The procedures are as follow:

```
procedure gradnamecombo;
begin
    sortplan_grad;
    sortplan_1grad2name;
    assign_seat;
    storeseat;
writeln('==========================');
    textcolor(green);
writeln('    Success !       ');
textcolor(white);
writeln('==========================');
    readln;
end;

procedure assign_seat;
var
  i : integer;
begin
  for i := 1 to stp do
     seat_num[i] := i;
end;

procedure storeseat;
var
  j : integer;
  s : text;
begin
 Clrscr;
 assign(s, 'seatinginfo.txt');
 rewrite(s);
   for j := 1 to stp do
   begin
   writeln(s, name[j]);
   writeln(s, age[j]);
   writeln(s, sex[j]);
   writeln(s, grad[j]);
   writeln(s, job[j]);
   writeln(s, seat_num[j]);
   end;
 close(s)
end;
```

After all the above action are done, user can view the seating plan freely. In this page, user can able to check total number of participants, which seat is not allocated will be shown with red color. In the lower part of the page, only the name according to their seating number will be shown with 4 rows based on the experience of user to check it quickly and conveniently. If necessary, its able to check each participants' information according to their seating number.

The procedure of the seating plan page:

```
procedure Splan;
var
  i,j,k,o,q,z,choice : integer;

begin
  readseat;
  Clrscr;
  i := 0;
  textcolor(yellow);
  writeln('Total participants : ', stp);
  textcolor(white);
  writeln;
  textcolor(green);
  writeln('              ABC School 50th Anniversary Celebration Dinner Seating Plan      ');
  textcolor(white);
  writeln;
  writeln;
  writeln;
  writeln('
_____________________________________________________________________________________________');
//16only
  write  ('    | ');
  for i:= 1 to 4 do         //first lane
  begin
      if i <= stp then
      begin
      textcolor(yellow);
      write  ('0',seat_num[i]);
      textcolor(white);
      write  (' | ');
      end
      else
      begin
      textcolor(red);
      write  (i);
      textcolor(white);
```

```
    write  (' | ');
    end;
end;

for i:= 9 to 12 do          //first lane
begin
    if i <= stp then
    begin
        if i = 9 then
        begin
        textcolor(yellow);
        write  ('0',seat_num[i]);
        textcolor(white);
        write  (' | ');
        end
        else
        begin
        textcolor(yellow);
        write  (seat_num[i]);
        textcolor(white);
        write  (' | ');
        end;
    end
    else
    begin
        if i = 9 then
         begin
         textcolor(red);
         write  ('0',i);
         textcolor(white);
         write  (' | ');
         end
        else
        begin
        textcolor(red);
        write  (i);
        textcolor(white);
        write  (' | ');
        end;
    end;
end;

for i:= 17 to 20 do          //first lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
```

```
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 25 to 28 do         //first lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
writeln;
writeln('   ------------------------------------------------------------------------------- ');
writeln('   |                                TABLE1                                |');
writeln('   ------------------------------------------------------------------------------- ');
write  ('    | ');
for i:= 5 to 8 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  ('0',seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
```

```
end;

for i:= 13 to 16 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 21 to 24 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 29 to 32 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
```

```
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
 end;
 writeln;
 writeln('
_____________________________________________________________________________________________');
 writeln;
 writeln;
 writeln('
_____________________________________________________________________________________________');
//16only
 write  ('    | ');
 for i:= 33 to 36 do         //third lane
 begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
 end;

 for i:= 41 to 44 do         //third lane
 begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
```

```
end;

for i:= 49 to 52 do         //third lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 57 to 60 do         //third lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
writeln;
writeln('   ------------------------------------------------------------------------------- ');
writeln('   |                                TABLE2                                |');
writeln('   ------------------------------------------------------------------------------- ');
write  ('    | ');
for i:= 37 to 40 do         //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
```

```
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 45 to 48 do         //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 53 to 56 do         //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 61 to 64 do         //forth lane
begin
```

```
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
writeln;
writeln('
_____________________________________________________________________________________________');
writeln;
writeln;
writeln;
j := 9;
k := 18;
o := 27;
q := 33;
writeln('========================================');
writeln('  Seating Numbers of the first table :  ');
writeln('========================================');
writeln;
for i:= 1 to 9 do
begin
    {if i < 10 then      //first display row (per nine)
    begin
    writeln('     0',seat_num[i],'         ',name[i]);
    end;

    if (i > 9) and (i < 20) then         //second display row
    begin
    writeln('     ',seat_num[i],'         ',name[i]);
    end;
    }
    write(' ',seat_num[i]:2,'  ',name[i]:10,'     ');
    write(' ',seat_num[i+j]:2,'  ',name[i+j]:10,'     ');
    write(' ',seat_num[i+k]:2,'  ',name[i+k]:10,'     ');
    if i+o < (stp/2)+1 then
    begin
    write(' ',seat_num[i+o]:2,'  ',name[i+o]:10,'     ');
    end;
    writeln;
```

```
  end;
  writeln;
  writeln('========================================');
  writeln('  Seating Numbers of the second table :  ');
  writeln('========================================');
  writeln;
  for z:= 0 to 8 do
  begin
      write(' ',seat_num[q+z],'  ',name[q+z]:10,'     ');
      write(' ',seat_num[q+z+j],'  ',name[q+z+j]:10,'     ');
      write(' ',seat_num[q+z+k],'  ',name[q+z+k]:10,'     ');
      if q+z+o < stp + 1 then
      begin
      write(' ',seat_num[q+z+o],'  ',name[q+z+o]:10,'     ');
      end;
      writeln;
  end;
  writeln();
  textcolor(yellow);
  writeln('                    1. Check personal information ');
  writeln('                    2. Return ');
  textcolor(white);
  write  ('                     Select(1-2): ');
  readln(choice);
  if (choice = 1) then
    begin
    ppi;
    end
  else
     begin
     inmenu := true;
     end
end;
```

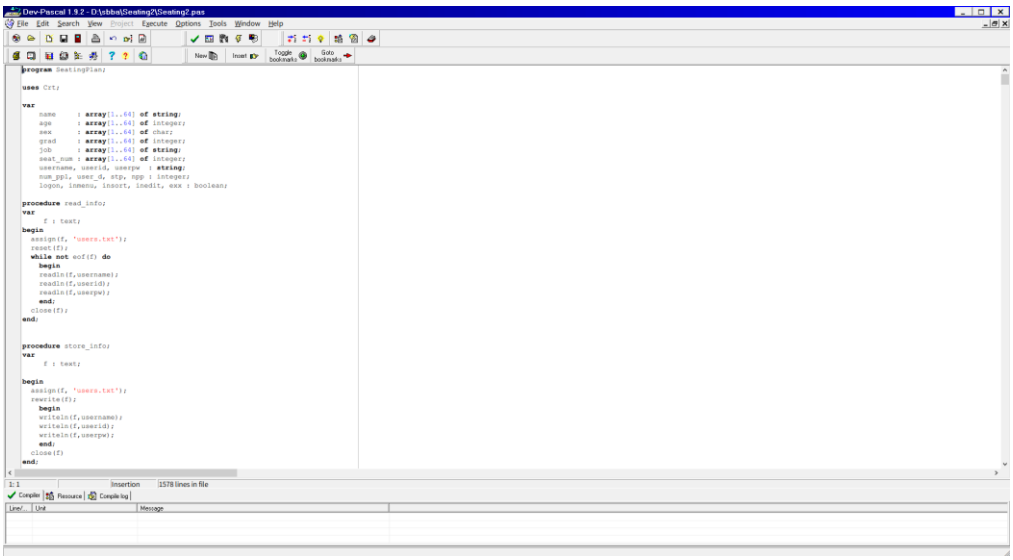
The procedure to search participants by seating number:

```
procedure ppi;
var
  i : integer;
  op : char;

begin
 writeln;
 write  ('              Please enter the number of seat you want to search : ');
 readln(npp);
 i := npp;
 if seat_num[i] < stp + 1 then
   begin
   writeln('              You are searching : ',seat_num[i]);
   writeln('              Name : ', name[i]);
   writeln('              Gender : ', sex[i]);
   writeln('              Age : ', age[i]);
   writeln('              Graduated in : ', grad[i]);
   writeln('              Employment : ', job[i]);
   writeln();
   end
 else
   begin
   textcolor(red);
   writeln('              No such person, please check again ');
   textcolor(white);
   writeln();
   end;
 write('              Continute to search ? (Y/N) : ');
 readln(op);
 if (op = 'N') then
    begin
    inmenu := true;
    end
 else
    begin
    ppi;
    end;
end;
```

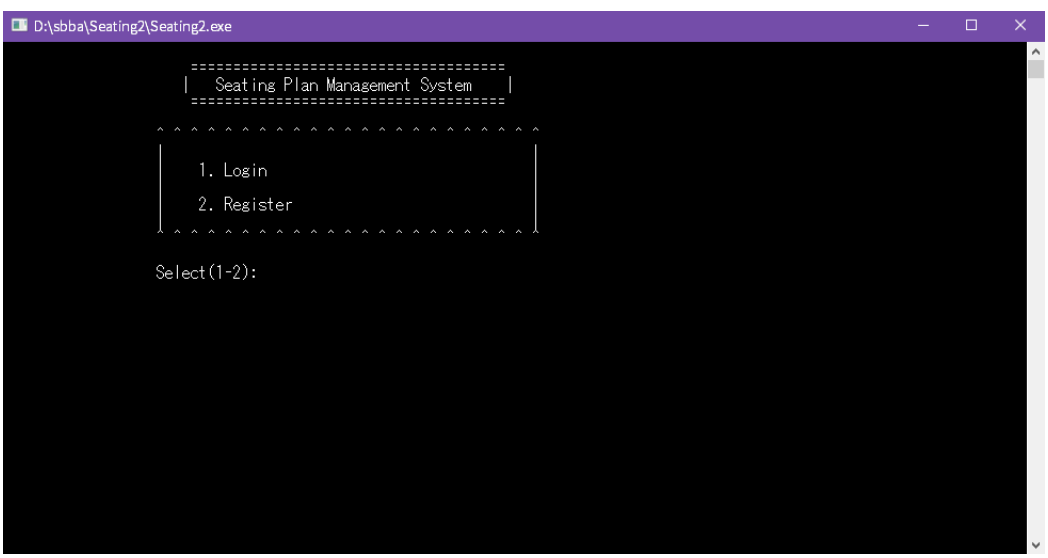
3.3 Program system

For this system, I choose Dev-Pascal as develop tool. Since the school-based syllabus is teaching this language, with the support of book and teaching material, I believe that these may facilitate my work. Also, the built-in compiler helps to convert my source program to executable file and save time.

The source program file is ‘Seating2.pas’.



The output executable file is ‘Seating2.exe’.



4.Testing & Evaluation

4.1 Outcome Error

a) Syntax Error

21 / 3	seating2.pas	Error: Identifier not found RESE
21 / 10	seating2.pas	Error: Illegal expression

Actually, with the warning while compile, this situation can be solved quickly. Since the application is able to direct me to where the error happened. For some typing error mostly are typo, like wrong spelling of the word or wrong variable. Also, symbols such as ; and := and = or wrong brackets for some conditional line. Meanwhile, begin and end expression is important for (e.g. if, else, for) sentence.

b) Run-time Error

I would say that this error isn't too trouble. Because I would test each procedure separately, and combine them part by part, just like assembling building block. Since each test only involve a few procedures, these errors can be erased soon.

c) Logic Error

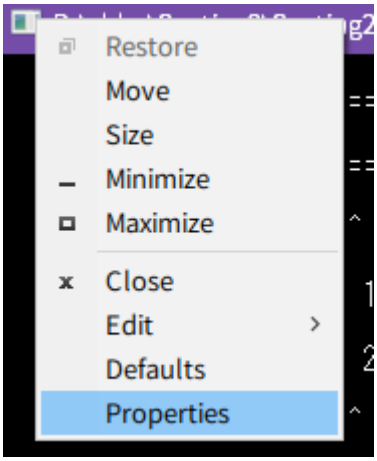
Since this type of error won't have any warning message, and it's the most time-consuming work. Because I have to check line by line, think about and dry run with a paper before coding. Also, I have to check different outcome for most of the possible input data. Showing some variable with words is needed because sometimes the value is not I am expected.

4.2 Testing the program

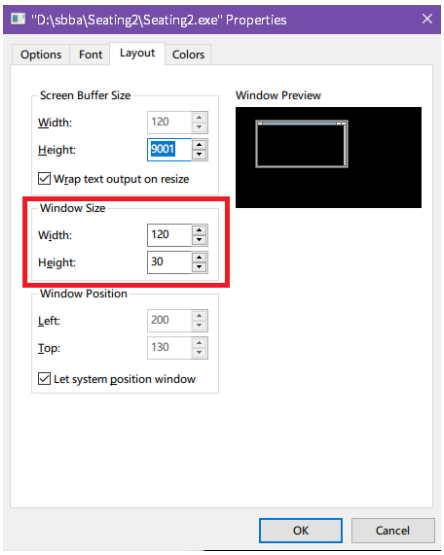
This program will be tested frequently for page to page, and focus on the error itself to prevent the relative event take place.

Remarks:

Its known that there will be a display issue due to the different resolution of monitor. For better experience, please read the instructions as following:



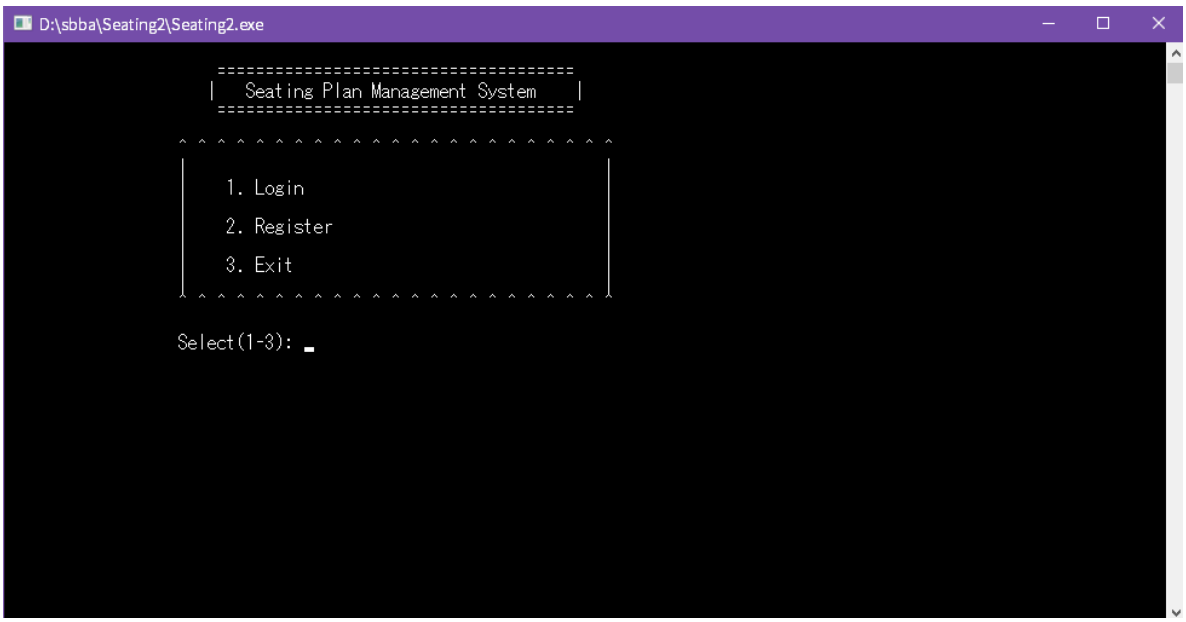
First, run the executable file(‘Seating2.exe’) and go on the left-top edge and right click, choose properties.



Then, go to the layout page and modify the value of width and height in the box of Window Size to 120 and 30 respectively. And click OK.

Showcase for the program:

1. First page



Input	Result

1	Proceed to the login function
2	Proceed to the register function
3	Close the window
Other integers(4,55,00,0)	Invalid, page refresh
Alphabet(a,n,qq,oh)	Run-time error (app crash)

Follow up action: Wait for further development. Problem of input alphabet still exist…

2. Registration page

```
Please enter your Name (at most 15 char) : asjidos_2323

Please enter your ID (4 char) : 882838a

The length of UserID must be 4!
Press <Enter> to retry.

Please enter your ID (4 char) : aa_3

Your UserID is aa_3

Please enter your password (at least 4 char) : aa

The length of password must be at least 4!
Press Enter to retry.

Your UserID is aa_3

Please enter your password (at least 4 char) : aaaa

Please enter your password again    : aaa

The passwords do not match!
Press Enter to retry.

Your UserID is aa_3

Please enter your password (at least 4 char) : aaaa

Please enter your password again    : aaaa

Success.
Press <Enter> to return. _
```

Name:

Input	Outcome
Anything less than 15 lengths	Pass
Larger 15 lengths	Invalid input, refresh page (G2.1)
Enter nothing	Invalid input, refresh page (G2.1)

(G2.1)

User ID:

Input	Outcome
Length 4	Pass
Less than 4	Invalid input, retry this step
Over 4	Invalid input, retry this step

Password:

Input	Outcome
Over or equal to 4	Pass
Less than 4	Invalid input, retry this step

Verify password:

Input	Outcome
Equal	Pass
Unequal	Invalid input, retry this step

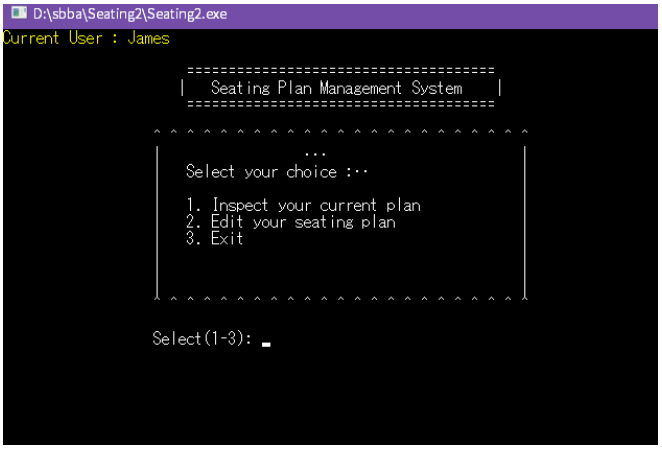
3. Login page

User ID & Password :

Input	Outcome
Correctly	Pass and proceed to the main menu
Wrongly	Invalid input, refresh page

Since this function will read the data in 'user.txt', only input the userid and password that the record is exist in the file can proceed to the next page.

4. Main menu



Input	Outcome

1	Proceed to the viewing page
2	Proceed to the edit menu
3	Close the window
Multiple digits number (22,333,40)	Refresh page
Enter nothing	Nothing happen
Alphabet	Run-time error, app crash

5. View page of seating plan

```
Total participants : 64
                ABC School 50th Anniversary Celebration Dinner Seating Plan

   | 01 | 02 | 03 | 04 | 09 | 10 | 11 | 12 | 17 | 18 | 19 | 20 | 25 | 26 | 27 | 28 |
   ---------------------------------------------------------------------------------
  |                                   TABLE1                                        |
   ---------------------------------------------------------------------------------
   | 05 | 06 | 07 | 08 | 13 | 14 | 15 | 16 | 21 | 22 | 23 | 24 | 29 | 30 | 31 | 32 |
   ---------------------------------------------------------------------------------

   | 33 | 34 | 35 | 36 | 41 | 42 | 43 | 44 | 49 | 50 | 51 | 52 | 57 | 58 | 59 | 60 |
   ---------------------------------------------------------------------------------
  |                                   TABLE2                                        |
   ---------------------------------------------------------------------------------
   | 37 | 38 | 39 | 40 | 45 | 46 | 47 | 48 | 53 | 54 | 55 | 56 | 61 | 62 | 63 | 64 |
   ---------------------------------------------------------------------------------

========================================
 Seating Numbers of the first table :
========================================

 1        Andy     10      Carmen     19       Elvis     28       Jena
 2    Angelica     11    Chrissie     20       Erick     29      Jonie
 3       Annie     12      Claire     21     Estella     30   Julianna
 4      Arthur     13       Clare     22        Gary     31      Kelley
 5       Asher     14       Colin     23      Grover     32        King
 6       Berny     15       Danny     24       Holly
 7       Bunny     16       Daryl     25       Issac
 8     Camilla     17        Davy     26       Jacob
 9        Carl     18      Elaina     27        Jean

========================================
 Seating Numbers of the second table :
========================================

 33     Kristin     42      Milton     51     Shelton     60      Trista
 34      Lavern     43      Norton     52     Stanley     61       Tyson
 35     Lorayne     44       Peter     53      Taylor     62       Velda
 36    Margaret     45        Phil     54     Terance     63        York
 37      Marina     46     Raymond     55      Terell     64         Zac
 38        Mary     47   Rosemarie     56        Teri
 39       Matty     48       Sally     57       Tobin
 40         Meg     49       Shane     58       Tonia
 41       Melba     50       Shawn     59        Tony

                      1. Check personal information
                      2. Return
                         Select(1-2): _
```

Input	Outcome
1	Proceed to the checking function
2	Back to main menu
Multiple digits number (22,1123)	Refresh page
Enter nothing	Nothing happen
Alphabet	Run-time error, app crash

5.1 Checking page

```
1. Check personal information
2. Return
   Select(1-2): 1

Please enter the number of seat you want to search :
```

Input	Outcome
1-64	Shows the detail information of that participant and ask to continue or not (G5.1.1)
0 or over 64	Invalid input, shows warning message and ask to continue or not (G5.1.2)

```
Please enter the number of seat you want to search : 1
You are searching : 1
Name : Andy
Gender : M
Age : 26
Graduated in : 2007
Employment : Employee

Continute to search ? (Y/N) :
```

(G5.1.1)

```
Please enter the number of seat you want to search : 100
No such person, please check again

Continute to search ? (Y/N) : _
```

(G5.1.2)

Input	Outcome
Y	Continue to the checking page
N	Back to the page of viewing seating plan
Other words or number	Invalid input, and back to main menu

5.2 Editing Page

```
Current User : James

                    ==================================
                   |   Seating Plan Management System    |
                    ==================================

                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
                |                  ...                          |
                |   Select your choice :··                      |
                |                                               |
                |   1. Make a new one                           |
                |   2. Insert a new person                      |
                |   3. Delete a person                          |
                |   4. Sort                                     |
                |   5. View participants                        |
                |   6. Return                                   |
                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^

                Select(1-6): _
```

Input	Outcome
1	Proceed to the function of adding new participants
2	Proceed to the function of inserting new participant(s)
3	Proceed to the function of deleting participant(s)
4	Proceed to the function of sorting participants
5	Proceed to the function of viewing current participants
6	Back to main menu
Alphabet	Run-time error, app crash

5.2.1 Adding new participants

```
Current participants :0 (Maximum 64)

Please enter the number of participants : _
```

```
Current participants :0 /64

Please enter the name : James
Please enter the age of James: 19
Please enter the sex of James: M
Please enter the year graduated of James: 1989
Please enter the employment of James: Employee_
```

(G5.2.1b)

```
Please enter the number of participants : 0
        Invalid input!
_
```

(G5.2.1c)

Input	Outcome
Number of participants: number 1-64	Pass and proceed to the next step (G5.2.1b)
0 or over 64	Invalid input, and return to the edit menu (G5.2.1c)
Alphabet	Run-time error, app crash

5.2.2 Insertion page

Condition 1:

```
Current participants :63 /64

Please enter the name : _
```

(Input page is same as 5.2.1)

```
Success !
```

(After input all the data of the participant)

Condition 2: You've got 64 participants entered.

```
All tables full !
You cannot add person anymore!!
```

And return to the edit page.

5.2.3 Delete a participant

```
Please enter the name of the person:
```

```
Please enter the name of the person: Q
No such person!
```

(Shows wrong message and return to edit page)

```
This person has been deleted.
```

(If input correctly and show the message before back to edit page)

5.2.4 Sorting page

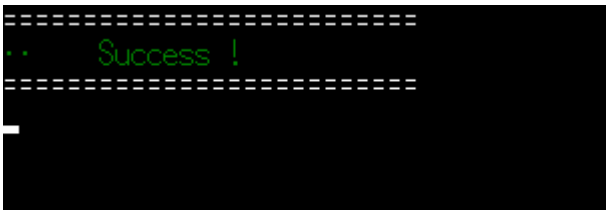
First option:

```
Current User : James

                    ====================================
                   |   Seating Plan Management System    |
                    ====================================

                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
                |                   ...                          |
                |   Select your first sorting way :              |
                |                                                |
                |   1. Sort by similar graduation year           |
                |   2. Balancing gender                          |
                |   3. Sort by name                              |
                |   4. Return                                    |
                |                                                |
                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^

                Select(1-4): _
```

Input	Outcome
1-3	Making the first choice for sorting, and proceed to the second option page
4	Return to the edit page
0 or over 4	Return to the edit page

Second option:

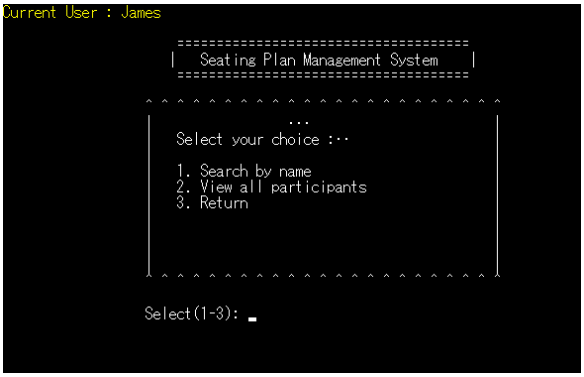
```
Current User : James

                    ====================================
                   |   Seating Plan Management System    |
                    ====================================

                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
                |                   ...                          |
                |   Select your second sorting way :             |
                |                                                |
                |   1. Sort by name                              |
                |   2. Balancing gender                          |
                |   3. Return                                    |
                |                                                |
                |                                                |
                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^

                Select(1-3): _
```



(G5.2.4a)

Input	Outcome
1-2	Making the second choice for sorting, then shows(G5.2.4a) before going back to the edit page
3	Go back to the first option of sorting page
0 or > 3	Back to the edit page

5.2.5 Viewing page



Option 1:

```
Current User : James

                    ====================================
                   |   Seating Plan Management System    |
                    ====================================

              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^
              |                  ...                        |
              |   Select your choice :··                    |
              |                                             |
              |   1. Search by name                         |
              |   2. View all participants                  |
              |   3. Return                                 |
              |                                             |
              |                                             |
              |                                             |
              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^

              Select(1-3): 1


            Please enter the name of the participant you want to search : _
```

```
Please enter the name of the participant you want to search : aa
No such person, please check again

Continute to search ? (Y/N) : _
```

(G5.2.5a)

```
Please enter the name of the participant you want to search : Teri
You are searching : Teri
Gender : F
Age : 27
Graduated in : 2005
Employment : Employee

Continute to search ? (Y/N) : _
```

(G5.2.5b)

Input	Outcome
Name not on data file or number	Invalid input, and shows the warning message, asking whether keep searching or back to the edit menu(G5.2.5a)
Correct name	Shows the detail information of that participant, and ask whether keep searching or back to the edit menu(G5.2.5b)

4.3 Self evaluation

Advantages:

It provides a suitable system for user. Including good looking, user-friendly interface and clear instruction. It's convenient to input and organize participants by each function. Also, I think this program is reusable, just need to adjust and modify the name of the event, and some of the requirements, such as table size, table amount, and maximum participants.

Disadvantages:

Apparently, any mistake on input data leads to run-time error and app crash. Affecting the whole fluency of the program. For instance, when I assign the input data type is integer, any alphabet or non-numeral typing are not allowed. Also, the table size is fixed, including the seats, unable to assign or change seat on my own desire, all rely on the sorting function.

Reflection:

For the disadvantages, I believe that its all caused by time-limiting and still having not enough ability to achieve them. As I keep learning and having enough experience of programming, I would like to achieve the above wills, and make even more function and delete the run-time error. Throughout this project, I found myself acquire more knowledge about coding, and strengthen my experience on it. Since this time is not like the working on textbook, I have to design the whole body but not only a few procedures.

Sometimes I am careless, because I frequently got syntax error for just not typing a symbol. Causing that the whole time for development is increased. But soon, when I get used to it, this error is erased gradually. At first, I have to compile a same program or procedure for several times just because of syntax error. Then I can able to compile successfully by only once

after a new procedure is done. Overall this experience is pretty funny, for we have to construct a whole program on our own. I can taste it when a complete program is born!

5. Reference and acknowledgement

From the internet

http://pascal-programming.info/lesson3.php

From textbooks

NSS Information and Communication Technology D1

NSS Information and Communication Technology D2

Acknowledgement

ICT Teacher-Mr.Chu

Friends for testing and suggestion

Appendix

Program source code (after testing & evaluation)

```
{
program SeatingPlan;

uses Crt;

var
        name     : array[1..64] of string;
        age      : array[1..64] of integer;
   sex      : array[1..64] of char;
        grad     : array[1..64] of integer;
        job      : array[1..64] of string;
   seat_num : array[1..64] of integer;
   username, userid, userpw  : string;
   num_ppl, user_d, stp, npp : integer;
   logon, inmenu, insort, inedit, exx, ex : boolean;

procedure read_info;
var
```

```
    f : text;
begin
  assign(f, 'user.txt');
  reset(f);
  while not eof(f) do
    begin
    readln(f,username);
    readln(f,userid);
    readln(f,userpw);
    end;
  close(f);
end;


procedure store_info;
var
    f : text;

begin
  assign(f, 'user.txt');
  rewrite(f);
    begin
    writeln(f,username);
```

```
    writeln(f,userid);
    writeln(f,userpw);
    end;
  close(f)
end;


procedure readpp;
var
  i : integer;
  p : text;

begin
  assign(p, 'participants.txt');
  reset(p);
  i := 0;
  while not eof(p) do
    begin
    i := i + 1;
    readln(p, name[i]);
    readln(p, age[i]);
    readln(p, sex[i]);
    readln(p, grad[i]);
```

```
    readln(p, job[i]);
    end;
  close(p);
  stp := i;
end;


procedure storepp;
var
  i : integer;
  p : text;

begin
  Clrscr;
  assign(p, 'participants.txt');
  rewrite(p);
  for i := 1 to num_ppl do
    begin
    writeln(p, name[i]);
    writeln(p, age[i]);
    writeln(p, sex[i]);
    writeln(p, grad[i]);
    writeln(p, job[i]);
```

```
    end;
  close(p)
end;

procedure store_newpp;
var
  i : integer;
  p : text;

begin
  Clrscr;
  assign(p, 'participants.txt');
  rewrite(p);
  for i := 1 to stp + 1 do
    begin
    writeln(p, name[i]);
    writeln(p, age[i]);
    writeln(p, sex[i]);
    writeln(p, grad[i]);
    writeln(p, job[i]);
    end;
  close(p)
end;
```

```
procedure store_mpp;
var
  i : integer;
  p : text;

begin
  Clrscr;
  assign(p, 'participants.txt');
  rewrite(p);
  for i := 1 to stp-1 do
    begin
    writeln(p, name[i]);
    writeln(p, age[i]);
    writeln(p, sex[i]);
    writeln(p, grad[i]);
    writeln(p, job[i]);
    end;
  close(p)
end;


procedure storeseat;
```

```
var
  j : integer;
  s : text;

begin
 Clrscr;
 assign(s, 'seatinginfo.txt');
 rewrite(s);
   for j := 1 to stp do
   begin
   writeln(s, name[j]);
   writeln(s, age[j]);
   writeln(s, sex[j]);
   writeln(s, grad[j]);
   writeln(s, job[j]);
   writeln(s, seat_num[j]);
   end;
 close(s)
end;

procedure readseat;
var
  s : text;
```

```
  j : integer;

begin
  assign(s, 'seatinginfo.txt');
  reset(s);
  j := 0;
    while not eof(s) do
    begin
    j := j + 1;
    readln(s, name[j]);
    readln(s, age[j]);
    readln(s, sex[j]);
    readln(s, grad[j]);
    readln(s, job[j]);
    readln(s, seat_num[j]);
    end;
  close(s)
end;


procedure create_acc;
var
  id1, user1, pw1, pw2 : string;
```

```
  id_b, pw_b, namepass : boolean;

begin
  clrscr;
  namepass := false;
  writeln;
  write('       Please enter your Name (at most 15 char) : ');
  readln(user1);
  if (length(user1) > 15) or (length(user1) = 0) then
    begin
    writeln;
    textcolor(red);
    writeln('                    Invalid Input ! ');
    textcolor(white);
    write  ('              Press <Enter> to retry. ');
    readln;
    create_acc;
    end
  else
  begin
  id_b := false;
  namepass := true;
  end;
```

```
if namepass = true then
begin
repeat
  writeln;
  writeln;
  write('       Please enter your ID (4 char) : ');
  readln(id1);
  if length(id1) <> 4 then
    begin
    writeln;
    textcolor(red);
    writeln('       The length of UserID must be 4!');
    textcolor(white);
    write('       Press <Enter> to retry. ');
    readln;
    end
  else
    begin
    id_b := true;
    end;
until id_b;
pw_b := false;
repeat
```

```
writeln;
writeln;
writeln('       Your UserID is ', id1);
writeln;
write('       Please enter your password (at least 4 char) : ');
readln(pw1);
if length(pw1) < 3 then
  begin
  writeln;
  textcolor(red);
  writeln('       The length of password must be at least 4!');
  textcolor(white);
  write('       Press Enter to retry. ');
  readln
  end
    else
      begin
        writeln;
        write('       Please enter your password again    : ');
        readln(pw2);
        if pw1 <> pw2 then
          begin
          writeln;
```

```
    textcolor(red);
    writeln('       The passwords do not match!');
    textcolor(white);
    write('       Press Enter to retry. ');
    readln
    end
  else
    begin
    user1 := user1 + '                        ';
    user1 := copy(user1, 1, 25);
    userid := id1;
    userpw := pw1;
    username := user1;
    store_info;
                pw_b := true;
    writeln;
    textcolor(green);
    writeln('       Success.');
    textcolor(white);
    write('       Press <Enter> to return. ');
    readln
    end
  end
```

```
  until pw_b
  end;
end;


procedure add_ppl;
var
   a, nump, age1, grad1 : integer;
   name1, job1 : string;
   sex1 : char;

begin
  Clrscr;
  inmenu := false;
  a := 0;
  num_ppl := 0;
  nump := 0;
  write('Current participants :');
  textcolor(yellow);
  write(a);
  textcolor(white);
  write(' ');
  write('(Maximum 64)');
```

```
writeln;
writeln;
writeln;
write('Please enter the number of participants : ');
readln(nump);
if (nump < 65) and (nump <> 0) then
begin
for a := 1 to nump do
  begin
  Clrscr;
  write('Current participants :');
  textcolor(yellow);
  write(a - 1);
  textcolor(white);
  write(' ');
  write('/', nump);
  writeln;
  writeln;
  writeln;
  write('Please enter the name : ');
  readln(name1);
  writeln;
  write('Please enter the age of ', name1, ': ');
```

```
  readln(age1);
  writeln;
  write('Please enter the sex of ', name1, ': ');
  readln(sex1);
  writeln;
  write('Please enter the year graduated of ', name1, ': ');
  readln(grad1);
  writeln;
  write('Please enter the employment of ', name1, ': ');
  readln(job1);
  writeln;
      name[a] := name1;
      age[a] := age1;
      sex[a] := sex1;
      grad[a] := grad1;
      job[a] := job1;
      num_ppl := num_ppl + 1;
      storepp;
  end;
end
else
begin
textcolor(red);
```

```
  writeln('      Invalid input!     ');
  textcolor(white);
  readln;
  inmenu := false;
  end;
end;


procedure del_ppl;
var
  i, j : integer;
  FindName : string;
begin
  Clrscr;
  inmenu := false;
  write('Please enter the name of the person: ');
  readln(FindName);
  j := 0;
  for i := 1 to stp do
    if name[i] = FindName
      then j := i;
  if j > 0 then
    begin
```

```
    for i := j to stp -1 do
          begin
          name[i] := name[i+1];
          age[i] := age[i+1];
    sex[i] := sex[i+1];
    grad[i] := grad[i+1];
    job[i] := job[i+1];
          end;
   store_mpp;
         writeln('This person has been deleted.');
   readln;
   stp := stp - 1
   end
  else
  begin
  textcolor(red);
  writeln('No such person!');
  textcolor(white);
  readln;
  end;
end;
```

```
procedure assign_seat;
var
  i : integer;

begin
  for i := 1 to stp do
     seat_num[i] := i;
end;

procedure sortplan_name;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
```

```
        if name[i] > name[i + 1]
          then begin
              temp_grad := grad[i];
              temp_age  := age[i];
              temp_name := name[i];
              temp_job  := job[i];
              temp_sex  := sex[i];
              grad[i]   := grad[i+1];
              age[i]    := age[i+1];
              name[i]   := name[i+1];
              job[i]    := job[i+1];
              sex[i]    := sex[i+1];
              grad[i+1] := temp_grad;
              age[i+1]  := temp_age;
              name[i+1] := temp_name;
              job[i+1]  := temp_job;
              sex[i+1]  := temp_sex;
              swapped := true;
              end;
  until (pass = stp - 1) or (not swapped)
end;
```

```
procedure sortplan_grad;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if grad[i] > grad[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
```

```
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;


procedure sortplan_1grad2name;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
```

```
pass := 0;
repeat
  pass := pass + 1;
  swapped := false;
  for i := 1 to (stp - pass) do
    if grad[i] = grad[i+1] then
    if name[i] > name[i + 1]
      then begin
          temp_grad := grad[i];
          temp_age  := age[i];
          temp_name := name[i];
          temp_job  := job[i];
          temp_sex  := sex[i];
          grad[i]   := grad[i+1];
          age[i]    := age[i+1];
          name[i]   := name[i+1];
          job[i]    := job[i+1];
          sex[i]    := sex[i+1];
          grad[i+1] := temp_grad;
          age[i+1]  := temp_age;
          name[i+1] := temp_name;
          job[i+1]  := temp_job;
          sex[i+1]  := temp_sex;
```

```
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;



procedure gradnamecombo;
begin
    sortplan_grad;
    sortplan_1grad2name;
    assign_seat;
    storeseat;
        writeln('==========================');
    textcolor(green);
        writeln('                    Success !      ');
        textcolor(white);
        writeln('==========================');
    readln;
end;




procedure sortplan_1name2gender;
```

```
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if name[i] > name[i+1] then
      if sex[i] <> sex[i + 1]
        then begin
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
```

```
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;


procedure namegendercombo;
begin
   sortplan_name;
   sortplan_1name2gender;
   assign_seat;
   storeseat;
        writeln('==========================');
   textcolor(green);
        writeln('                    Success !       ');
```

```
        textcolor(white);
        writeln('==========================');
    readln;
end;



procedure sortplan_1grad2gender;
var
  pass, i, temp_age, temp_grad : integer;
  temp_name, temp_job : string;
  temp_sex : char;
  swapped : boolean;

begin
  Clrscr;
  pass := 0;
  repeat
    pass := pass + 1;
    swapped := false;
    for i := 1 to (stp - pass) do
      if grad[i] = grad[i+1] then
      if sex[i] <> sex[i + 1]
        then begin
```

```
            temp_grad := grad[i];
            temp_age  := age[i];
            temp_name := name[i];
            temp_job  := job[i];
            temp_sex  := sex[i];
            grad[i]   := grad[i+1];
            age[i]    := age[i+1];
            name[i]   := name[i+1];
            job[i]    := job[i+1];
            sex[i]    := sex[i+1];
            grad[i+1] := temp_grad;
            age[i+1]  := temp_age;
            name[i+1] := temp_name;
            job[i+1]  := temp_job;
            sex[i+1]  := temp_sex;
            swapped := true;
            end;
  until (pass = stp - 1) or (not swapped)
end;

procedure gradgendercombo;
   begin
   sortplan_grad;
```

```
    sortplan_1grad2gender;
    assign_seat;
    storeseat;
        writeln('==========================');
    textcolor(green);
        writeln('                    Success !      ');
        textcolor(white);
        writeln('==========================');
    readln;
    end;


procedure displayppl;
var i : integer;
begin
    Clrscr;
    inmenu := false;
    sortplan_name;
    writeln('          List of participants:         ',' (',stp,') ');
    writeln;
    writeln('          Name       Age   Gender');
    writeln('================================');
    textcolor(yellow);
```

```
    for i:= 1 to stp do
        begin
        writeln('  ',i:2,' : ',name[i]:9, age[i]:5, sex[i]:5);
        end;
    textcolor(white);
    readln;
end;

procedure SortGrad;
var
  choice1 : integer;

begin
  Clrscr;
  textcolor(yellow);
  writeln('Current User : ', username);
  textcolor(white);
  writeln;
  writeln('                    ====================================        ');
  writeln('                    |   Seating Plan Management System    |        ');
  writeln('                    ====================================        ');
  writeln;
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
```

```
  writeln('                |                                          |      ');
  writeln('                |   Select your second sorting way :       |      ');
  writeln('                |                                |      ');
  writeln('                |   1. Sort by name                    |      ');
  writeln('                |   2. Balancing gender                 |      ');
  writeln('                |   3. Return                        |      ');
  writeln('                |                                |      ');
  writeln('                |                                |      ');
  writeln('                |                                |      ');
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln;
  write('                Select(1-3): ');
  readln(choice1);
  writeln;
  writeln;
     case choice1 of
     1 : gradnamecombo;
     2 : gradgendercombo;
     //3 :
     end;
end;

procedure SortGender;
```

```
var
  choice1 : integer;

begin
  Clrscr;
  textcolor(yellow);
  writeln('Current User : ', username);
  textcolor(white);
  writeln;
  writeln('                  ====================================        ');
  writeln('                  |   Seating Plan Management System    |        ');
  writeln('                  ====================================        ');
  writeln;
  writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
  writeln('              |                                          |      ');
  writeln('              |   Select your second sorting way :       |      ');
  writeln('              |                                  |      ');
  writeln('              |   1. Sort by similar graduation year      |      ');
  writeln('              |   2. Sort by name                    |      ');
  writeln('              |   3. Return                          |      ');
  writeln('              |                                  |      ');
  writeln('              |                                  |      ');
  writeln('              |                                  |      ');
```

```
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln;
  write('                Select(1-3): ');
  readln(choice1);
  writeln;
  writeln;
     case choice1 of
     1 : gradgendercombo;
     2 : namegendercombo;
     //3 : insort := true;
     end;
end;


procedure SortName;
var
  choice1 : integer;

begin
  Clrscr;
  textcolor(yellow);
  writeln('Current User : ', username);
  textcolor(white);
```

```
writeln;
writeln('                ====================================        ');
writeln('                |   Seating Plan Management System    |        ');
writeln('                ====================================        ');
writeln;
writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
writeln('              |                                          |     ');
writeln('              |   Select your second sorting way :       |     ');
writeln('              |                                  |     ');
writeln('              |   1. Sort by similar graduation year     |     ');
writeln('              |   2. Balancing gender                    |     ');
writeln('              |   3. Return                          |     ');
writeln('              |                                  |     ');
writeln('              |                                  |     ');
writeln('              |                                  |     ');
writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
writeln;
write('              Select(1-3): ');
readln(choice1);
writeln;
writeln;
   case choice1 of
   1 : gradnamecombo;
```

```
    2 : namegendercombo;
    end;
end;




procedure sorts;
var
  choice1 : integer;

begin
  if length(name[1]) < 1 then
  begin
  textcolor(red);
  writeln('        You must add participants first ! ');
  textcolor(white);
  readln;
  inmenu := false;
  end
  else
  begin
  Clrscr;
  inmenu := false;
```

```
textcolor(yellow);
writeln('Current User : ', username);
textcolor(white);
writeln;
writeln('                ====================================        ');
writeln('                |   Seating Plan Management System    |        ');
writeln('                ====================================        ');
writeln;
writeln('            ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
writeln('            |                                          |     ');
writeln('            |   Select your first sorting way :        |     ');
writeln('            |                                |     ');
writeln('            |   1. Sort by similar graduation year      |     ');
writeln('            |   2. Balancing gender                  |     ');
writeln('            |   3. Sort by name                    |     ');
writeln('            |   4. Return                        |     ');
writeln('            |                                |     ');
writeln('            |                                |     ');
writeln('            ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
writeln;
write('            Select(1-4): ');
readln(choice1);
writeln;
```

```
  writeln;
     case choice1 of
     1 : SortGrad;
     2 : SortGender;
     3 : SortName;
     //4 :
     end;
  end;
end;

procedure backtomenu;
begin
 inedit := false;
 insort := false;
 inmenu := true;
end;

procedure insert_ppl;
var
  k,age1,grad1 : integer;
  name1, job1 : string;
  sex1 : char;
  f : text;
```

```
begin
  Clrscr;
  inmenu := false;
  assign(f,'participants.txt');
  if stp = 64 then
  begin
  textcolor(red);
  writeln;
  writeln;
  writeln('                All tables full !           ');
  writeln('        You cannot add person anymore!!  ');
  textcolor(white);
  readln;
  end
  else
  begin
    read_info;
    Clrscr;
    k:=stp+1;
    write('Current participants :');
    textcolor(yellow);
    write(stp);
```

```
textcolor(white);
write(' ');
write('/64');
writeln;
writeln;
writeln;
write('Please enter the name : ');
readln(name1);
writeln;
write('Please enter the age of ', name1, ': ');
readln(age1);
writeln;
write('Please enter the sex of ', name1, ': ');
readln(sex1);
writeln;
write('Please enter the year graduated of ', name1, ': ');
readln(grad1);
writeln;
write('Please enter the employment of ', name1, ': ');
readln(job1);
writeln;
    append(f);
    name[k] := name1;
```

```
        age[k] := age1;
        sex[k] := sex1;
        grad[k] := grad1;
        job[k] := job1;
        num_ppl := num_ppl + 1;
        close(f);
        store_newpp;
   textcolor(green);
   writeln('        Success !         ');
   textcolor(white);
   readln;
  end
end;

procedure searchppl;
var
  i : integer;
  op : char;
  nametp : string;
  found : boolean;

begin
  writeln;
```

```
 write   ('                    Please enter the name of the participant you want to
search : ');
 readln(nametp);
 found := false;
 i := 1;
 while (i < stp +1) and (not found) do
  begin
  if nametp = name[i] then
    begin
    found := true;
    end
    else
    begin
    found := false;
    i := i +1;
    end;
  end;
  if (found = true) then
   begin
   writeln('              You are searching : ',name[i]);
   writeln('              Gender : ', sex[i]);
   writeln('              Age : ', age[i]);
   writeln('              Graduated in : ', grad[i]);
```

```
  writeln('              Employment : ', job[i]);
  writeln();
  end
  else
  begin
  textcolor(red);
  writeln('              No such person, please check again ');
  textcolor(white);
  writeln();
  end;
write('              Continute to search ? (Y/N) : ');
readln(op);
if (op = 'N') then
  begin
  inmenu := false;
  inedit := true;
  end
else
  begin
  searchppl;
  end;
```

```
end;


procedure views;
var
  choice : integer;
begin
  Clrscr;
  inedit := true;
  read_info;
  readpp;
  textcolor(yellow);
  writeln('Current User : ', username);
  textcolor(white);
  writeln;
  writeln('                  ====================================        ');
  writeln('                  |   Seating Plan Management System    |        ');
  writeln('                  ====================================        ');
  writeln;
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln('                |                                          |      ');
  writeln('                |   Select your choice :                   |      ');
  writeln('                |                                  |      ');
```

```
  writeln('                |   1. Search by name                    |      ');
  writeln('                |   2. View all participants             |      ');
  writeln('                |   3. Return                        |      ');
  writeln('                |                                |      ');
  writeln('                |                                |      ');
  writeln('                |                                |      ');
  writeln('                |                                |      ');
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln;
  write('                Select(1-3): ');
  readln(choice);
  writeln;
  writeln;
     case choice of
     1 : searchppl;
     2 : displayppl;
     3 : inmenu := false;
     end;
end;



procedure edits;
var
```

```
  choice : integer;
begin
  Clrscr;
  inedit := true;
  read_info;
  readpp;
  textcolor(yellow);
  writeln('Current User : ', username);
  textcolor(white);
  writeln;
  writeln('                  ====================================        ');
  writeln('                  |   Seating Plan Management System    |        ');
  writeln('                  ====================================        ');
  writeln;
  writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln('              |                                          |      ');
  writeln('              |   Select your choice :                   |      ');
  writeln('              |                                |      ');
  writeln('              |   1. Make a new one                |      ');
  writeln('              |   2. Insert a new person           |      ');
  writeln('              |   3. Delete a person               |      ');
  writeln('              |   4. Sort                      |      ');
  writeln('              |   5. View participants             |      ');
```

```
  writeln('                |   6. Return                        |      ');
  writeln('                |                                |      ');
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln;
  write('                Select(1-6): ');
  readln(choice);
  writeln;
  writeln;
     case choice of
     1 : add_ppl;
     2 : insert_ppl;
     3 : del_ppl;
     4 : sorts;
     5 : views;
     6 : backtomenu;
     end;
end;



procedure ppi;
var
   i : integer;
   op : char;
```

```
begin
  writeln;
  write  ('              Please enter the number of seat you want to search : ');
  readln(npp);
  i := npp;
  if (seat_num[i] < stp + 1) and (seat_num[i] <> 0) then
    begin
    writeln('              You are searching : ',seat_num[i]);
    writeln('              Name : ', name[i]);
    writeln('              Gender : ', sex[i]);
    writeln('              Age : ', age[i]);
    writeln('              Graduated in : ', grad[i]);
    writeln('              Employment : ', job[i]);
    writeln();
    end
  else
    begin
    textcolor(red);
    writeln('              No such person, please check again ');
    textcolor(white);
    writeln();
    end;
```

```
  write('              Continute to search ? (Y/N) : ');
  readln(op);
  if (op = 'N') then
    begin
    inmenu := true;
    end
  else if (op ='Y') then
    begin
    ppi;
    end
  else
    begin
    writeln('          Invalid Input ');
    readln;
    end;



end;



procedure Splan;
var
  i,j,k,o,q,z,choice : integer;
```

```
begin
  readseat;
  Clrscr;
  i := 0;
  textcolor(yellow);
  writeln('Total participants : ', stp);
  textcolor(white);
  writeln;
  textcolor(green);
  writeln('                    ABC School 50th Anniversary Celebration Dinner Seating
Plan      ');
  textcolor(white);
  writeln;
  writeln;
  writeln;
  writeln('
_____________________________________________________________________________
____________________');  //16only
  write  ('    | ');
  for i:= 1 to 4 do         //first lane
  begin
      if i <= stp then
      begin
```

```
    textcolor(yellow);
    write  ('0',seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 9 to 12 do          //first lane
begin
    if i <= stp then
    begin
        if i = 9 then
        begin
        textcolor(yellow);
        write  ('0',seat_num[i]);
        textcolor(white);
```

```
    write  (' | ');
    end
    else
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end;
end
else
begin
    if i = 9 then
     begin
     textcolor(red);
     write  ('0',i);
     textcolor(white);
     write  (' | ');
     end
    else
    begin
    textcolor(red);
    write  (i);
```

```
        textcolor(white);
        write  (' | ');
        end;
    end;
end;

for i:= 17 to 20 do         //first lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
```

```
for i:= 25 to 28 do         //first lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
writeln;
writeln('   ------------------------------------------------------------------------------- ');
writeln('   |                                TABLE1                                |');
writeln('   ------------------------------------------------------------------------------- ');
write  ('    | ');
```

```
for i:= 5 to 8 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  ('0',seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 13 to 16 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
```

```
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 21 to 24 do         //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
```

```
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 29 to 32 do          //second lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
```

```
    end;
  end;
  writeln;
  writeln('
_____________________________________________________________________
____________________');
  writeln;
  writeln;
  writeln('
_____________________________________________________________________
____________________');  //16only
  write  ('    | ');
  for i:= 33 to 36 do         //third lane
  begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
```

```
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 41 to 44 do         //third lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
```

```
for i:= 49 to 52 do         //third lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 57 to 60 do         //third lane
begin
    if i <= stp then
    begin
```

```
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
writeln;
writeln('   ------------------------------------------------------------------------------- ');
writeln('   |                                TABLE2                                |');
writeln('   ------------------------------------------------------------------------------- ');
write  ('    | ');
for i:= 37 to 40 do         //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
```

```
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 45 to 48 do         //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
```

```
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;

for i:= 53 to 56 do          //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
```

```
    end;
end;

for i:= 61 to 64 do         //forth lane
begin
    if i <= stp then
    begin
    textcolor(yellow);
    write  (seat_num[i]);
    textcolor(white);
    write  (' | ');
    end
    else
    begin
    textcolor(red);
    write  (i);
    textcolor(white);
    write  (' | ');
    end;
end;
writeln;
```

```
writeln('
_____________________________________________________________________
____________________');
writeln;
writeln;
writeln;
j := 9;
k := 18;
o := 27;
q := 33;
writeln('========================================');
writeln('  Seating Numbers of the first table :  ');
writeln('========================================');
writeln;
for i:= 1 to 9 do
begin
    {if i < 10 then      //first display row (per nine)
    begin
    writeln('     0',seat_num[i],'          ',name[i]);
    end;

    if (i > 9) and (i < 20) then          //second display row
    begin
```

```
    writeln('     ',seat_num[i],'          ',name[i]);
    end;
    }
    write(' ',seat_num[i]:2,'  ',name[i]:10,'     ');
    write(' ',seat_num[i+j]:2,'  ',name[i+j]:10,'     ');
    write(' ',seat_num[i+k]:2,'  ',name[i+k]:10,'     ');
    if i+o < (stp/2)+1 then
    begin
    write(' ',seat_num[i+o]:2,'  ',name[i+o]:10,'     ');
    end;
    writeln;
end;
writeln;
writeln('========================================');
writeln('  Seating Numbers of the second table :  ');
writeln('========================================');
writeln;
for z:= 0 to 8 do
begin
    write(' ',seat_num[q+z],'  ',name[q+z]:10,'     ');
    write(' ',seat_num[q+z+j],'  ',name[q+z+j]:10,'     ');
    write(' ',seat_num[q+z+k],'  ',name[q+z+k]:10,'     ');
    if q+z+o < stp + 1 then
```

```
      begin
      write(' ',seat_num[q+z+o],'  ',name[q+z+o]:10,'     ');
      end;
      writeln;
  end;
  writeln();
  textcolor(yellow);
  writeln('                    1. Check personal information ');
  writeln('                    2. Return ');
  textcolor(white);
  write  ('                     Select(1-2): ');
  readln(choice);
  if (choice = 1) then
    begin
    ppi;
    end
  else
     begin
     inedit := false;
     end
end;
```

```
procedure menu;
var
  choice : integer;
begin
  Clrscr;
  inmenu := true;
  textcolor(yellow);
  writeln('Current User : ', username);
  textcolor(white);
  writeln;
  writeln('                  ====================================        ');
  writeln('                  |   Seating Plan Management System    |        ');
  writeln('                  ====================================        ');
  writeln;
  writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln('              |                                          |      ');
  writeln('              |   Select your choice :                   |      ');
  writeln('              |                                |      ');
  writeln('              |   1. Inspect your current plan          |      ');
  writeln('              |   2. Edit your seating plan             |      ');
  writeln('              |   3. Exit                        |      ');
  writeln('              |                                |      ');
```

```
  writeln('                |                              |      ');
  writeln('                |                              |      ');
  writeln('                ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^      ');
  writeln;
  write('                Select(1-3): ');
  readln(choice);
  writeln;
  writeln;
     case choice of
     1 :  Splan;
     2 :  edits;
     3 :  exx := true;
     end;
end;


procedure loginsys(var user_d : integer);
var
  id, pw : string;
  found : boolean;
  i : integer;

begin
```

```
clrscr;
writeln;
writeln;
writeln('                    ====================================        ');
writeln('                    |   Seating Plan Management System    |        ');
writeln('                    ====================================        ');
writeln;
writeln('               ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
writeln('               |                                |      ');
writeln('               |              LOGIN              |      ');
writeln('               |                                |      ');
writeln('               ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
writeln;
write('                    UserID : ');
readln(id);
writeln;
write('                   Password : ');
readln(pw);
writeln;
writeln;
found := false;
i := 0;
while (not found) and (i < 1) do
```

```
  begin
    i := i + 1;
    if (id = userid) and (pw = userpw) then
      begin
        found := true;
        user_d := 1;
        logon := true;
      end
  end;
 if (not found) then
  begin
    user_d := 0;
    textcolor(red);
    writeln(':20,'> > > Invalid UserID or Password!');
    write(':20,'> > > Press <Enter> to return.');
    textcolor(white);
    logon := false;
    readln
  end
 else
    menu;

end;
```

```
procedure loginh;
var
  choice : integer;
begin
   clrscr;
   textcolor(white);
   writeln;
   writeln('                  ====================================        ');
   writeln('                 |   Seating Plan Management System    |        ');
   writeln('                  ====================================        ');
   writeln;
   writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
   writeln('              |                                |      ');
   writeln('              |    1. Login                      |      ');
   writeln('              |                                |      ');
   writeln('              |    2. Register                    |      ');
   writeln('              |                                |      ');
   writeln('              |    3. Exit                      |      ');
   writeln('              |                                |      ');
   writeln('              ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^ ^     ');
   writeln;
```

```
  write('                Select(1-3): ');
  readln(Choice);
  writeln;
  case choice of
    1 : loginsys(user_d);
    2 : create_acc;
    3 : exx := true;
  end;
end;

begin
     read_info;
     readpp;

     inmenu := false;
     inedit := false;
     insort := false;
     exx := false;
     ex  := false;
      repeat
           if (logon = false) then
           loginh;
      until logon;
```

```
    repeat
        if (inedit = false) and (inmenu = true) then
        menu;
        if (inmenu = false) and (inedit = true) then
        edits;

    until exx;

end.
}
```

Working schedule

Date	Task
June-2017	Choice of Topic, Background research + Define the objectives + Propose Functions
Aug-2017	Design of solution
Sep-2017	Implementation
Nov-2017	Testing & Evaluation, Conclusion & Discussion + Final Report