Hong Kong Diploma of Secondary Examination 201x

Information and Communication Technology

School-based Assessment (SBA)

Module D

Software Development

Seating Plan

Content

Introduction

Our 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. During the data collection stage, the personal information of the participants will be inputted into the program, as shown as below:

- Name of participants
- Year of graduation of participants
- Sex of participants
- Elective of participants
- Number of seats required From Participants

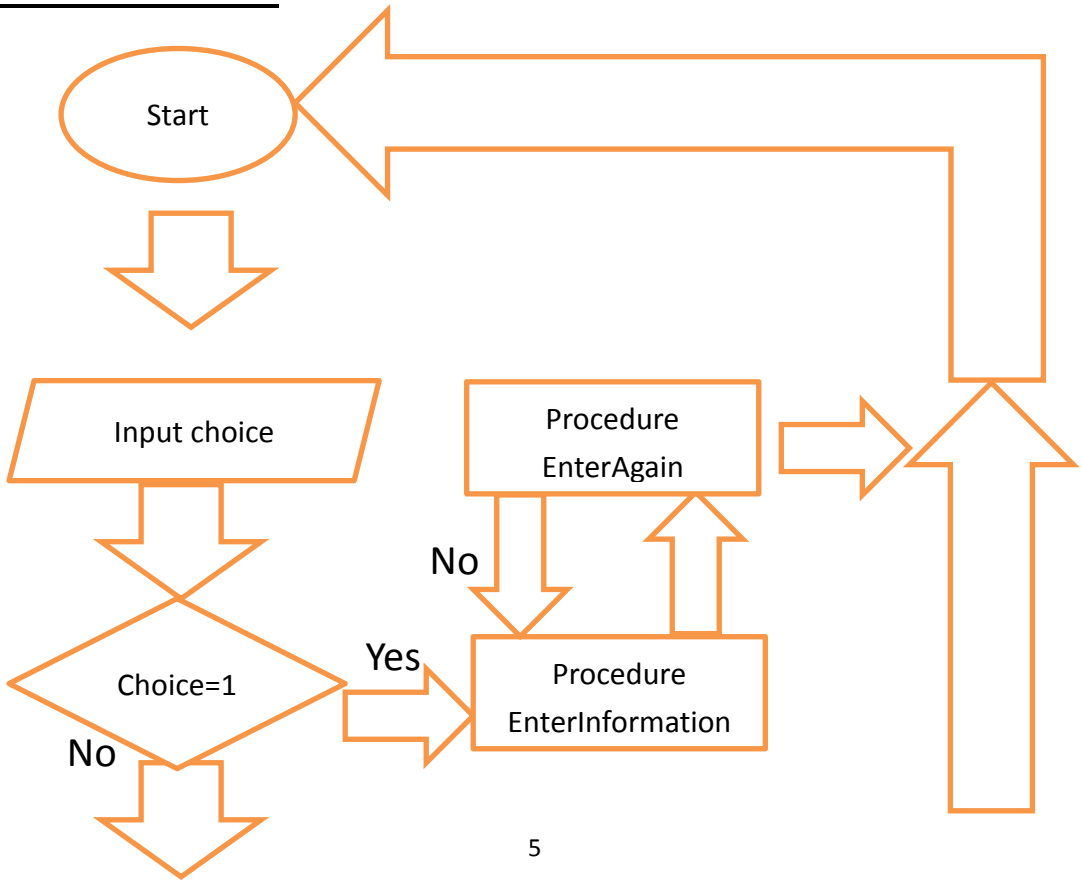
Design

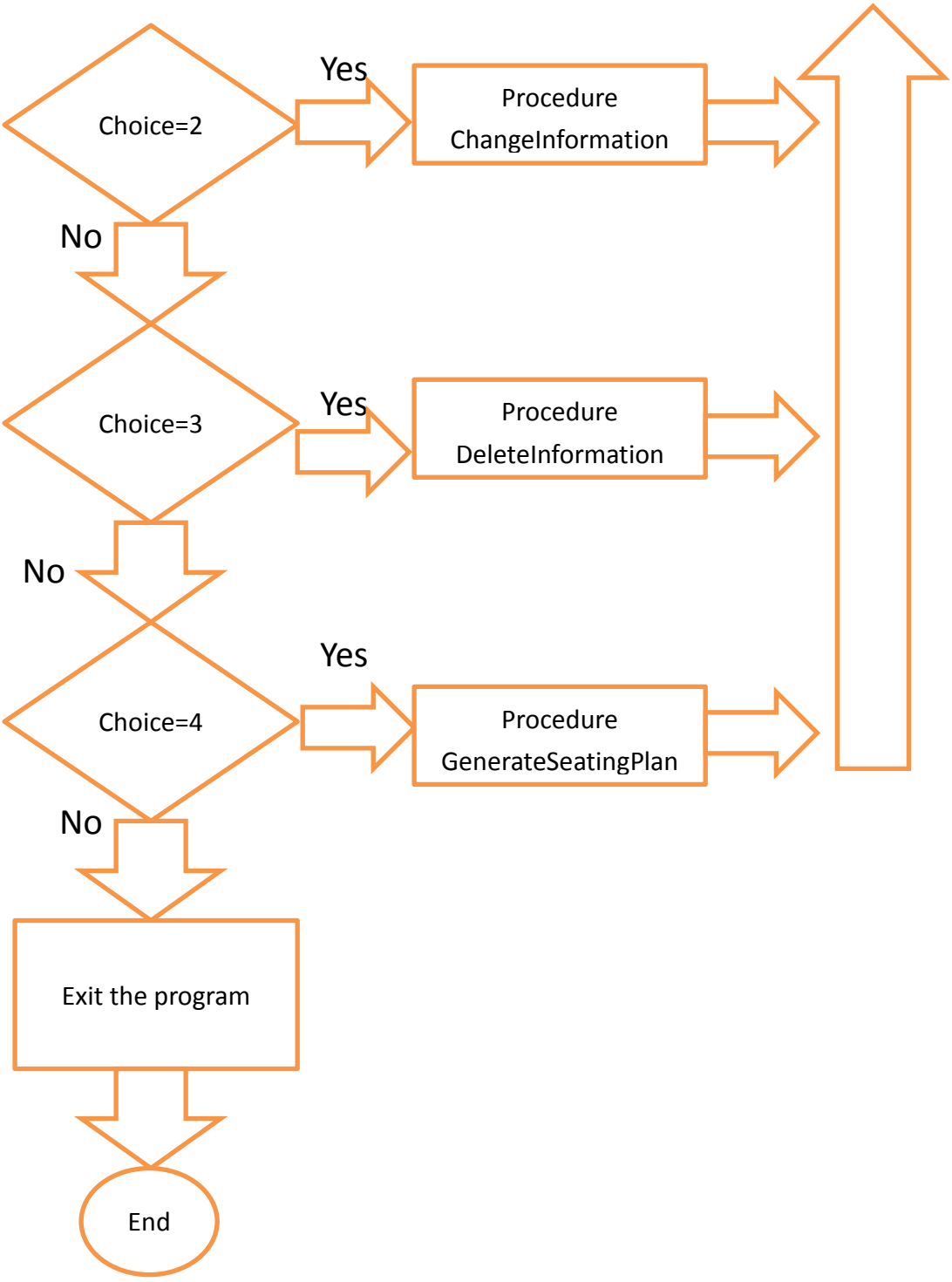
User Interface

The Command Line Interface (CLI) is used. The advantages of using CLI are that it can be much faster than any other type of interface if the user knows the correct commands. It also requires much less memory (RAM) in order to use compared to other types of user interfaces.

There is no multimedia elements are added to the program.

Modularization





Data Structure

name: string, for participants to enter their name.

year: integer, for participants to enter their year of graduation.

sex: character, for participants to enter their sex.

elective: integer, for participants to enter their elective.

seat: Integer, for participants to enter the number of seat they required.

data: text, used to read data file.

a ,b: integer

numberofpeople: integer, to record the number of people.

readseat: array[1..secondmax] of integer, to read the number of seats required from the participants.

top: integer, for reading array, data

quarry: integer, for participants to enter their seat code.

counter: integer, used to check the Boolean.

found: Boolean, to check whether the change of the information is correct.

readseatcode: array[1..secondmax] of integer, to read the seat code of the participants.

stay: integer, for reading the data from the array.

readname: array[1..secondmax] of string, to read the name of the participants.

readyear: array[1..secondmax] of integer, to read the year of graduation of the participants.

readsex: array[1..secondmax] of char, to read the sex of the participants.

readelective: array[1..secondmax] of integer, to read the elective of the participants.

d: text, for reading data.txt

e: integer, for reading data in array.

code: integer, for participants to enter their seat code.

loaddata: array[1..secondmax] of integer, for reading the seat code of each participants.

deletecode: integer, for reading the data from the array which are going to delete.

f,g,h,j,k,l,m,n,o: integer, for the use of rading data from different groups that divided.

hold: integer, used for the loops of reading data from array.

group1data, group2data, group3data, group4data, group5data, group6data, group7data, group8data, group9data: array[1..secondmax], for reading data from the group.

group1number, group2number, group3number, group4number, group5number, group6number, group7number, group8number, group9number: integer, to save the number of participants from the group.

space: integer, to record, check the number of seat.

table: read, to record, check the number of table.

group1sit, group2sit, group3sit, group4sit, group5sit, group6sit, group7sit, group8sit, group9sit: integer, to record the number of participants sits in the table in group.

group1finish, group2finish, group3finish, group4finish, group5finish, group6finish, group7finish, group8finish, group9finish: Boolean, to check whether the participants in group has all been sit.

numberofparticipant: Integer, to store the number of the participants sited in the table.

finaltable: integer, to store the final number of table.

numberoftable: integer, to store the number of the table used.

tabledata: array[1..secondmax,1..10] of integer, to load the data, number of participants in table.

tablename: array[1..secondmax,1..10] of integer, to read the participants' name of the table.

tablesex: array[1..econdmax, 1..10 of char, to record participant's sex in the table.

tableelective: array[1..secondmax, 1..10] of integer, to record participants' elective in the table.

tableyear: array[1..secondmax, 1..10] of integer, to record participants' year of graduation in the table.

i: string, for user to enter 'next' to return to main menu.

search: integer, used in different array for searching each of the participants' information.

full: Boolean, to check whether the number of participants in table is full.

sit: Boolean, to check whether there is a participant sited alone.

table number: integer, to show the number of table among all the table.

stop: Boolean, to change whether the display, generation of seating plan is finished.

require: integer, to load the seat required from the participants.

finalnumber: array[1..secondmax] of integer, used to check the final number of participants sit in the table.

fgroup, ggoup,

hgroup,jgroup,kgroup,lgroup,mgroup,ngroup,ogroup: array[1..secondmax] of Boolean, to check whether the people in the group are arranged to the table.

tablefull: array[1..secondmax] of Boolean, to check whether the table is full of the participants.

quarryfound: Boolean, to check whether the seat code is existed.

loaddatause: array[1..secondmax] of Boolean, to check whether the information of the participant is right.

Constants and Limitations

Secondmax=500

The only constant used in the program is secondmax. The maximum data amount handled in the program is 500.

Data Control

To reduce the input errors, the follow procedure are used to control the data input.

Data validation:

Procedure Name_Check:

It is used to check the input range of the participant's name. The name entered out of the length of 25 strings will be disallowed. Length check is used in this procedure

Procedure YearOfGraduation_Check:

It is used to check the input range of the participant's year of graduation. The year of graduation entered out of the range of 1977-2017 will be diallowed. Range check s used in this procedure.

Procedure Sex_Check:

It is used to check the input of the participant's sex. The sex entered not 'M' or 'F' will be disallowed.

Procedure Elective_Check:

It is used to check the input range of the participant's elective. The elective entered out of the range of 1-3 will be disallowed.

Range check s used in this procedure.

Procedure Seat_Check:

It is used to change the input of participant’s number of seats required. The seats required out of the range 1-5 will be disallowed. Range check s used in this procedure.

Data verification:

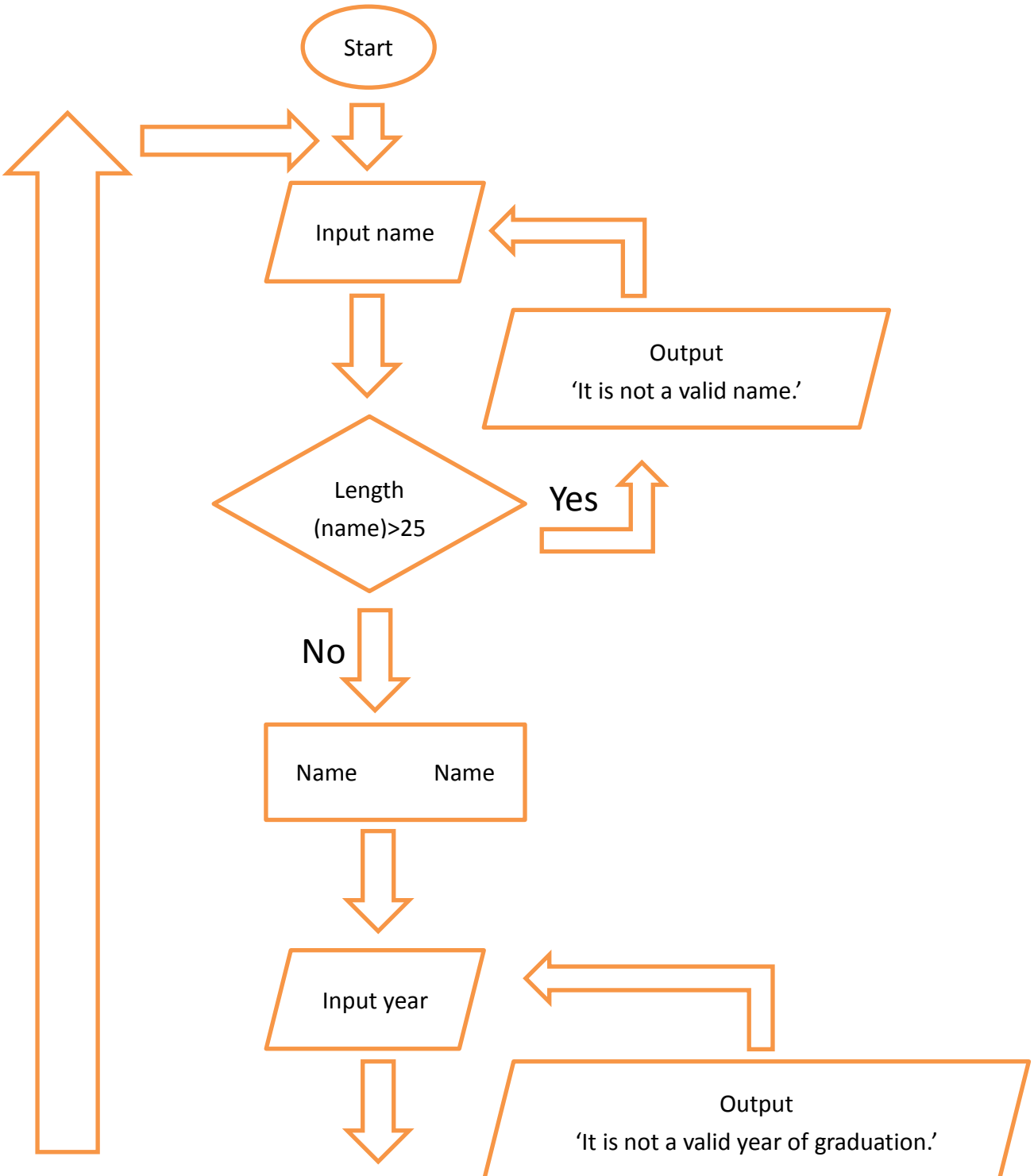
Procedure EnterAgain:

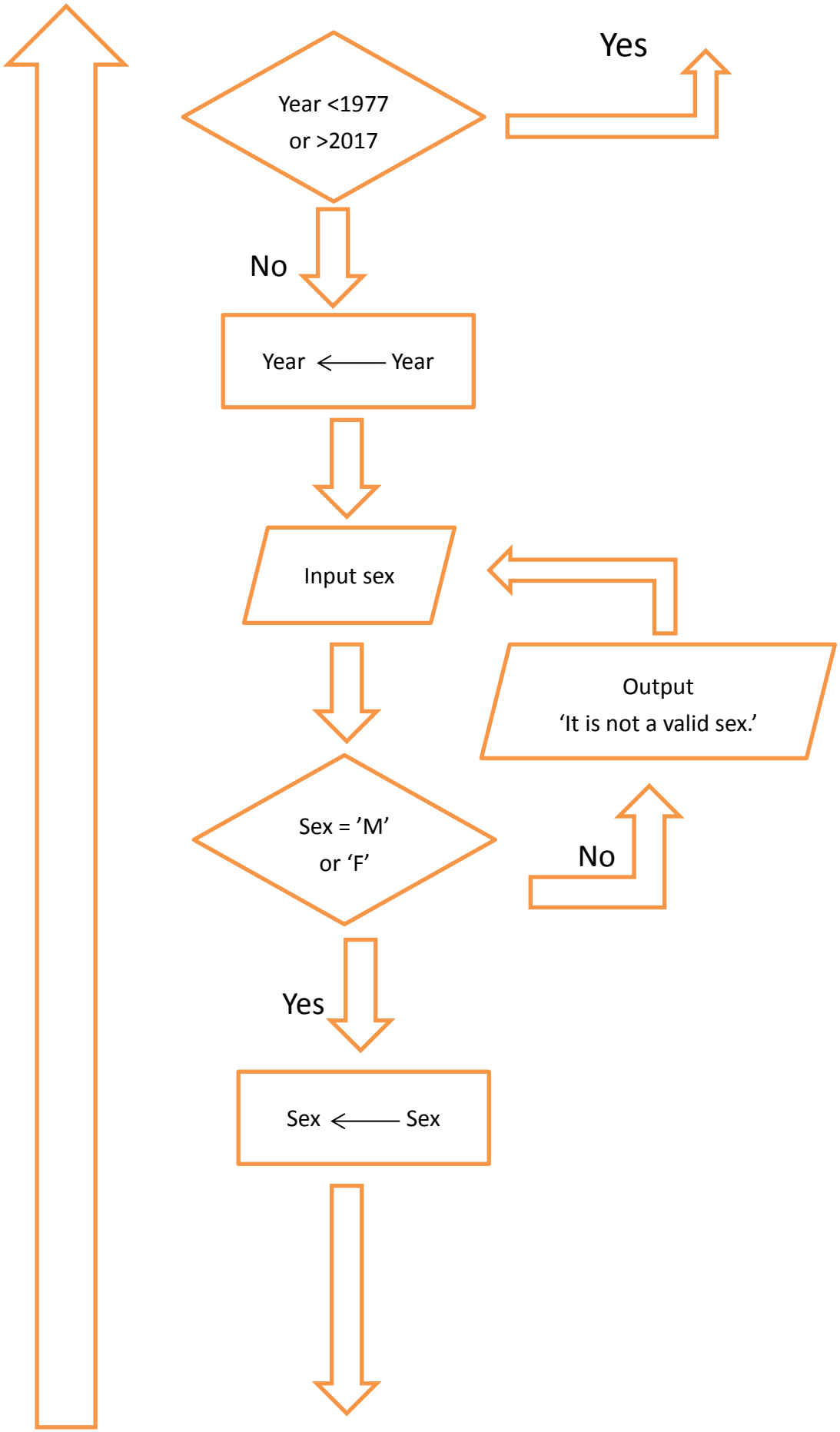
It is used for participants who are not sure or confused about the data they entered recently. It provides them a second chance to enter their personal information, which is one of the methods of data verification-double entry that entering the data twice and comparing the two copies.

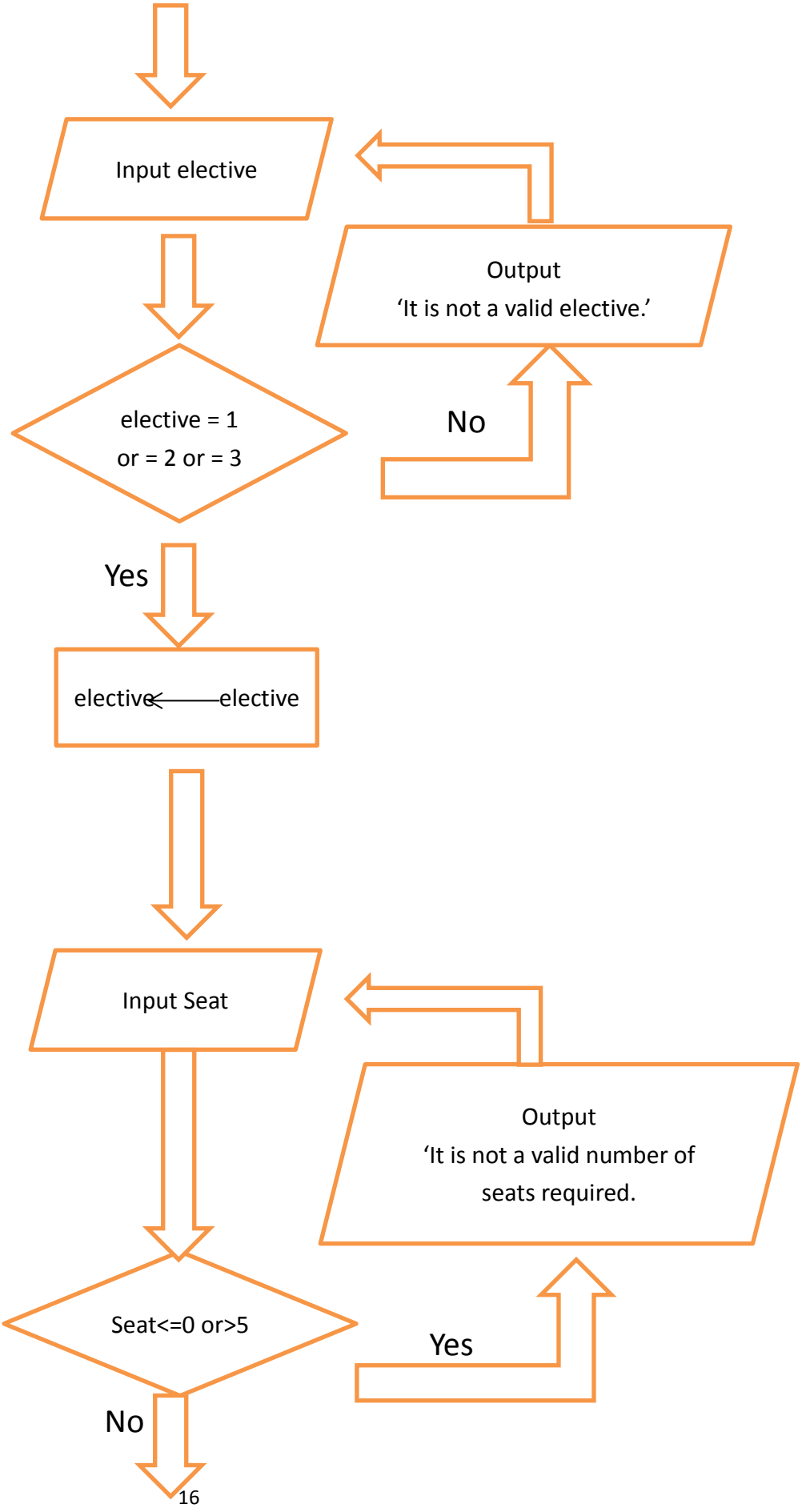
Implementation

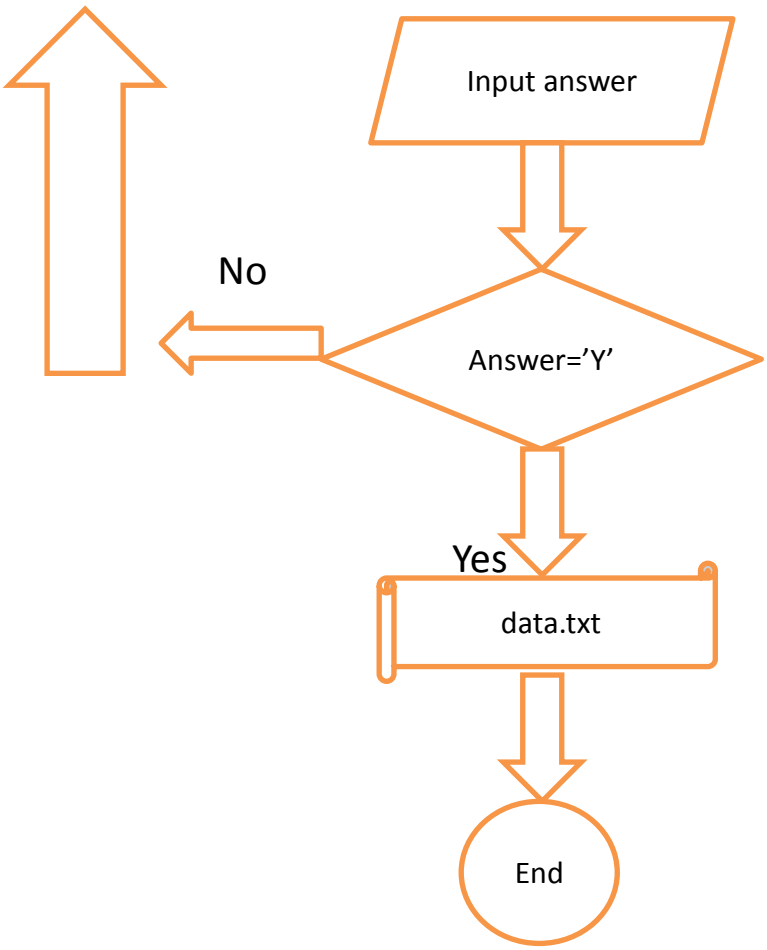
Operations of Each Module

Procedure EnterInformation:

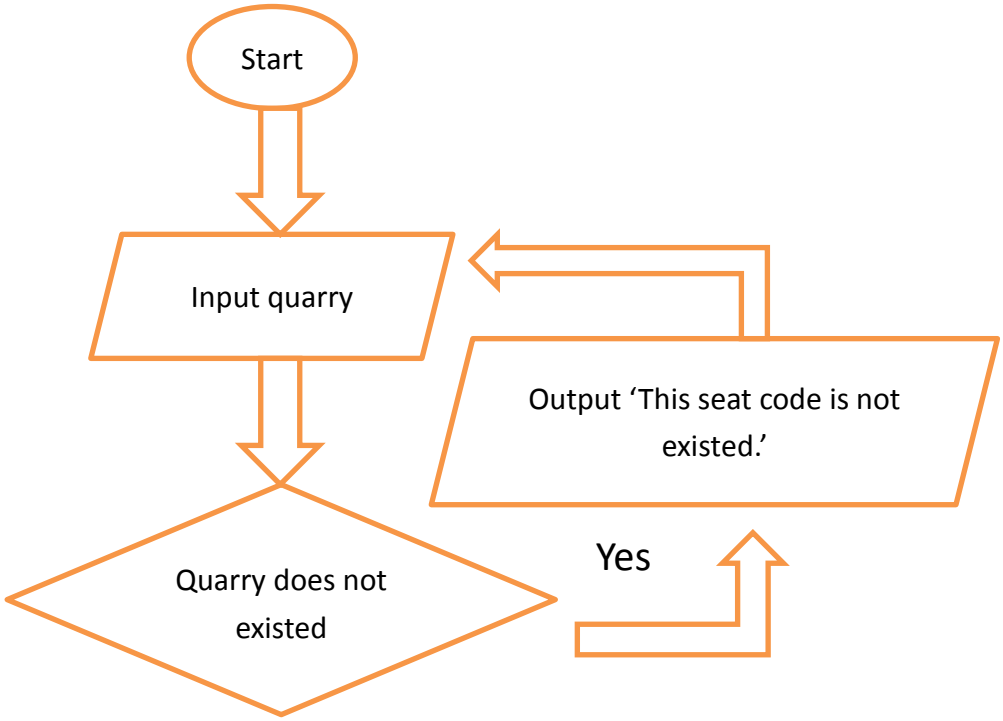


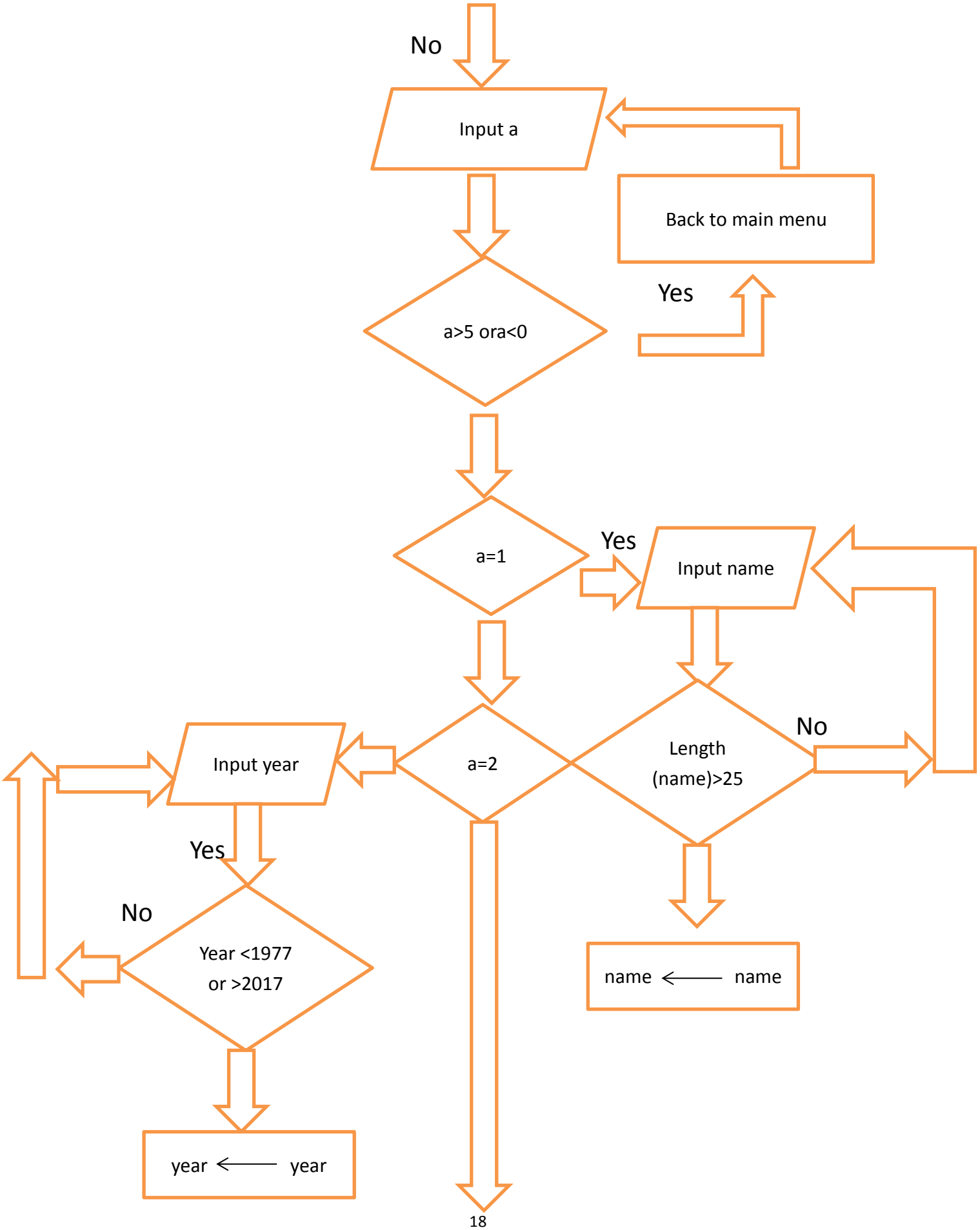

Year <1977
or >2017
Yes
No
Year ← Year
Input sex
Output
'It is not a valid sex.'
Sex = 'M'
or 'F'
No
Yes
Sex ← Sex

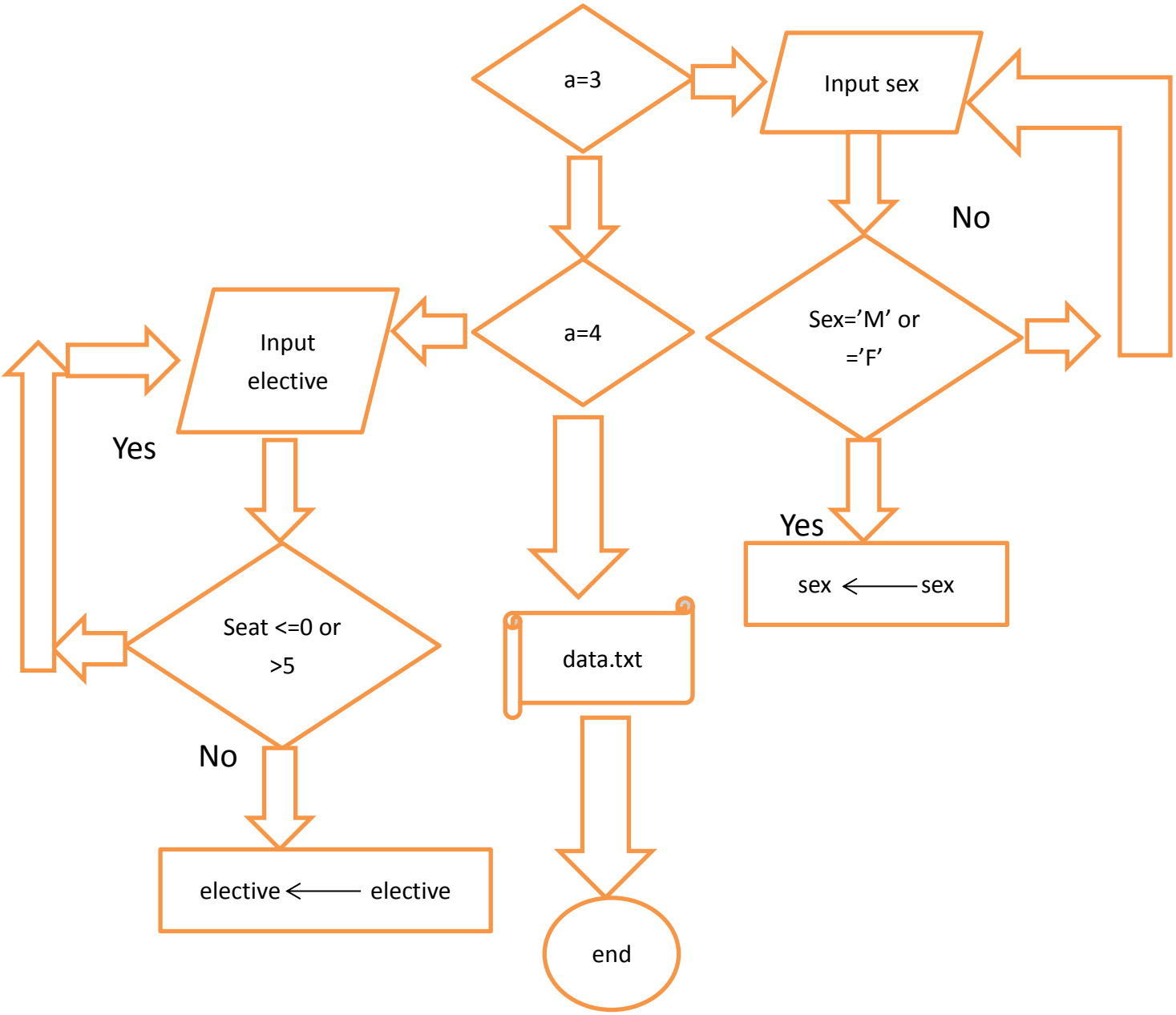

Input elective
Output
'It is not a valid elective.'
elective = 1
or = 2 or = 3
No
Yes
elective←elective
Input Seat
Output
'It is not a valid number of
seats required.
Seat<=0 or>5
Yes
No



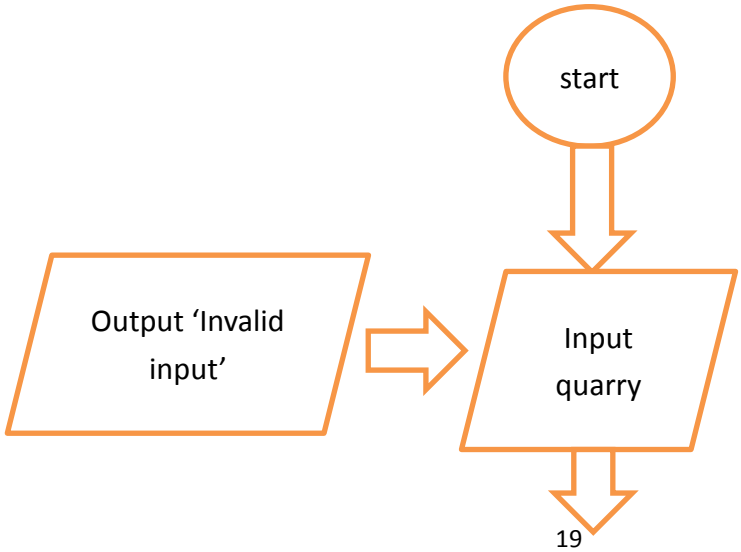
Procedure ChangeInformation

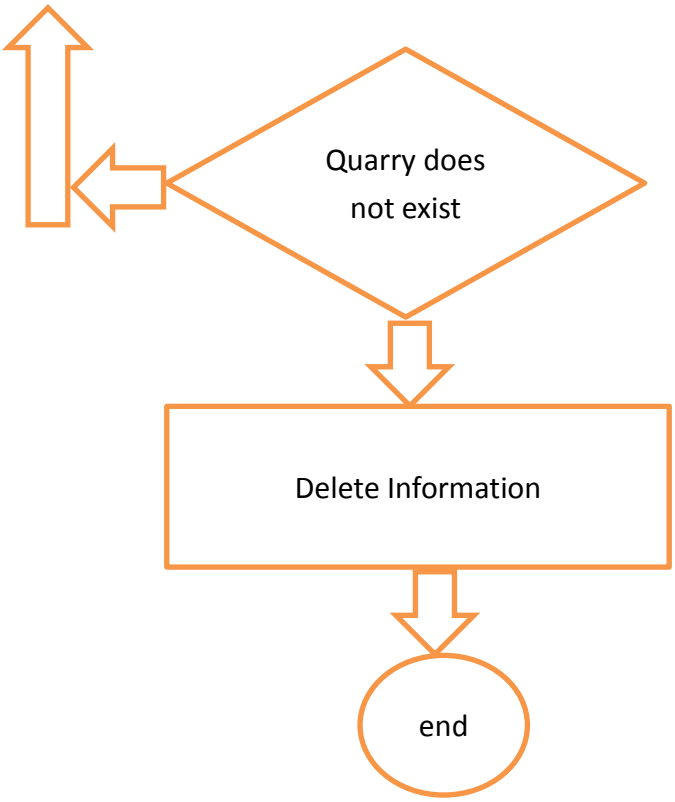



No
Input a
Back to main menu
Yes
a>5 ora<0
a=1
Yes
Input name
No
Length
(name)>25
a=2
Input year
Yes
No
Year <1977
or >2017
name ← name
year ← year

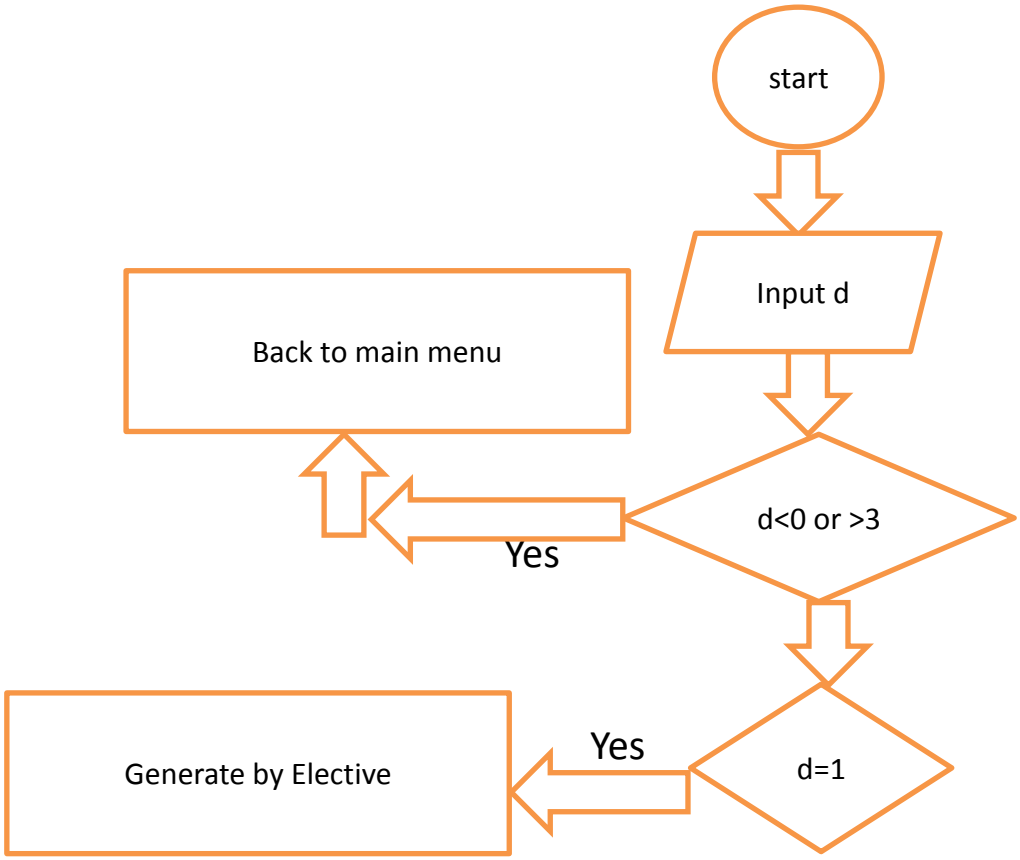


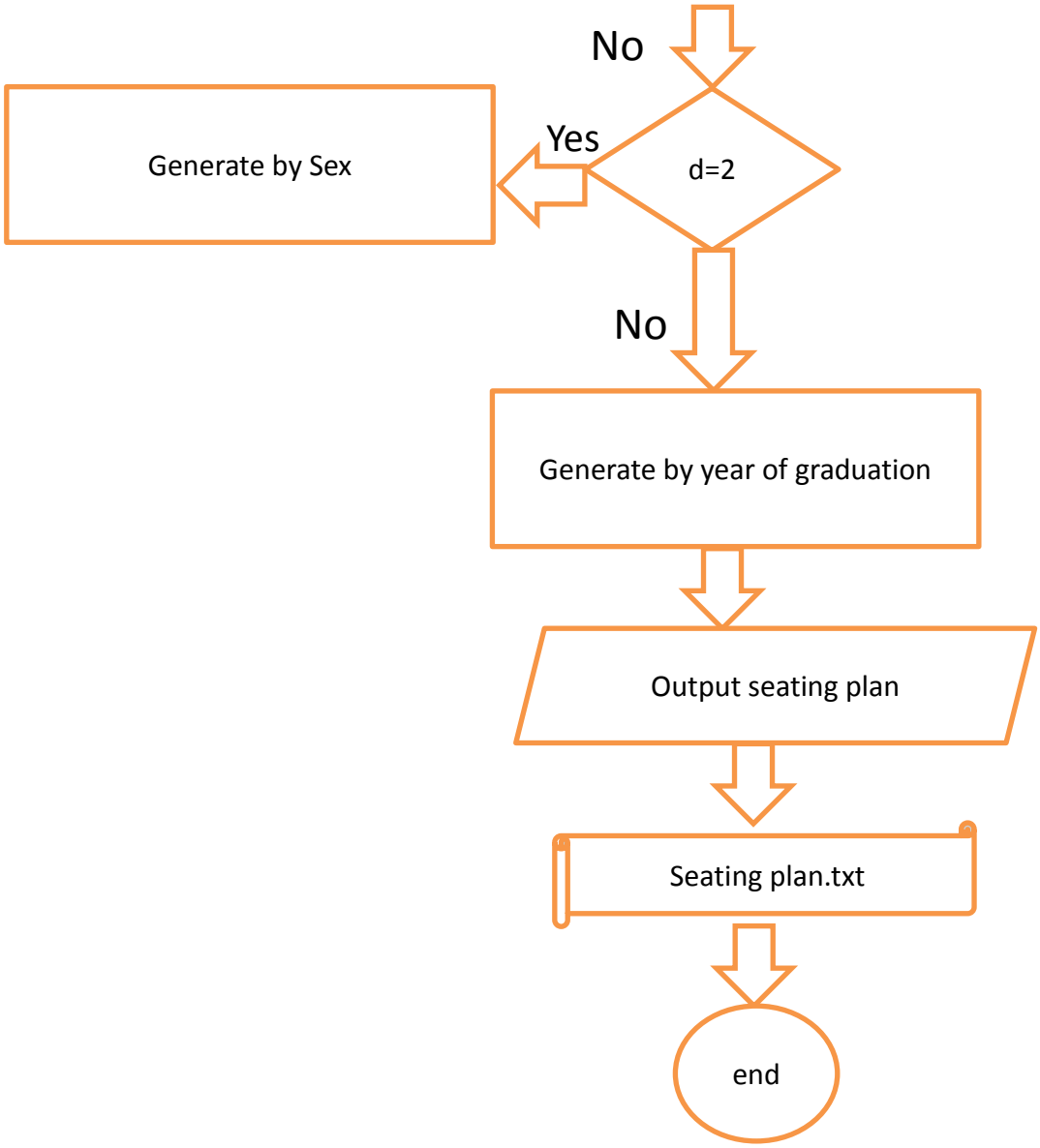
Procedure DeleteInformation



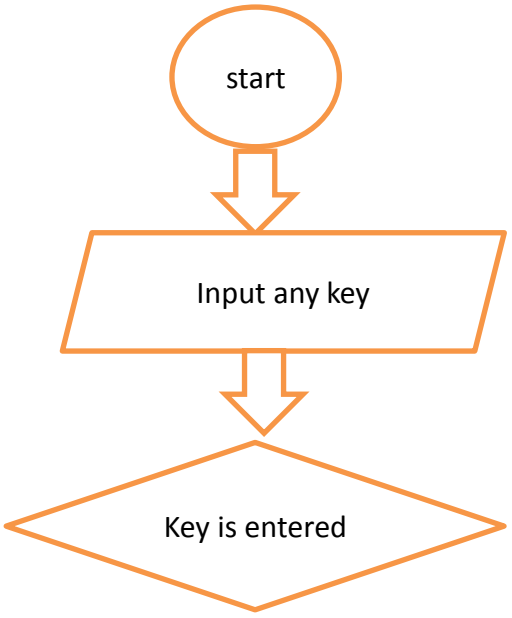


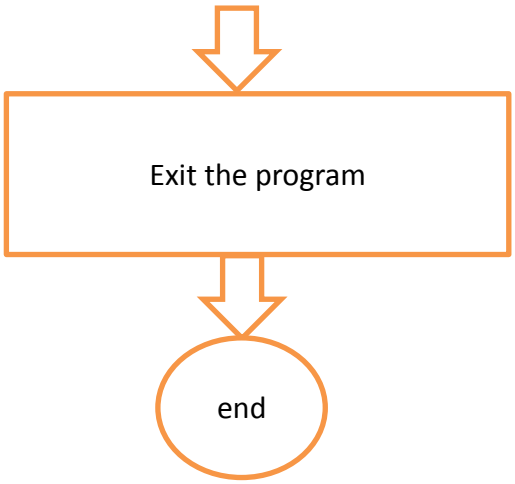
Procedure GenerateSeatingPlan





Procedure Exitprogram





Algorithms applied

Insertion sort is used in the program for generating the seating plan and writing the data into the file.

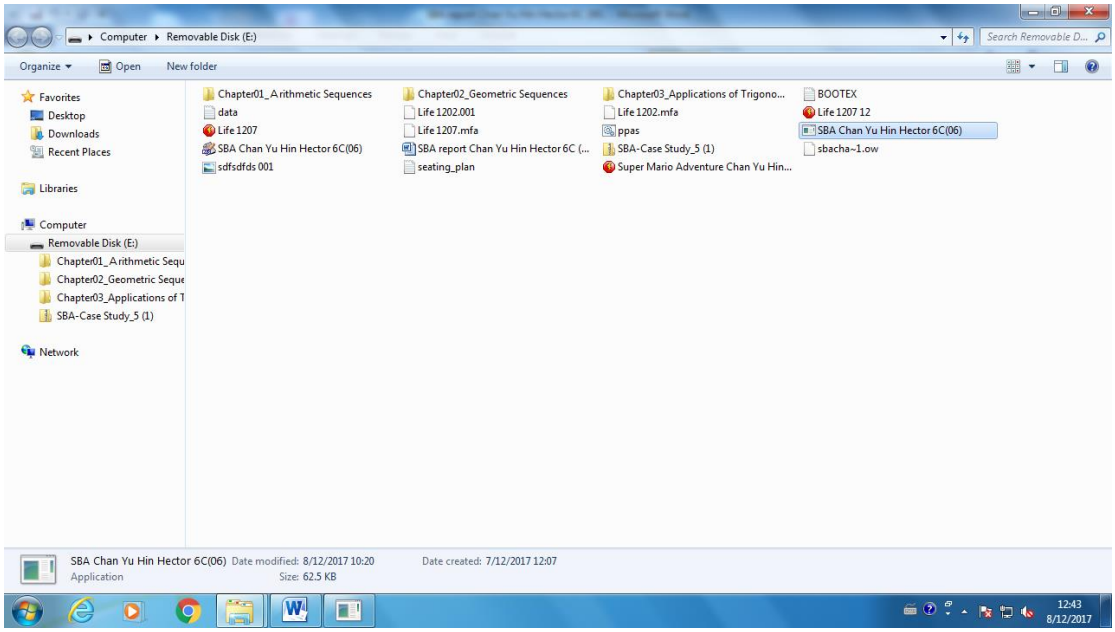
Insertion sort is a simple sorting algorithm that builds the final sorted array (or list) one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, or merge sort. However, insertion sort provides several advantages:

- Simple implementation
- Efficient for (quite) small data sets, much like other quadratic sorting algorithms
- More efficient in practice than most other simple quadratic algorithms such as selection sort or bubble sort

User guide

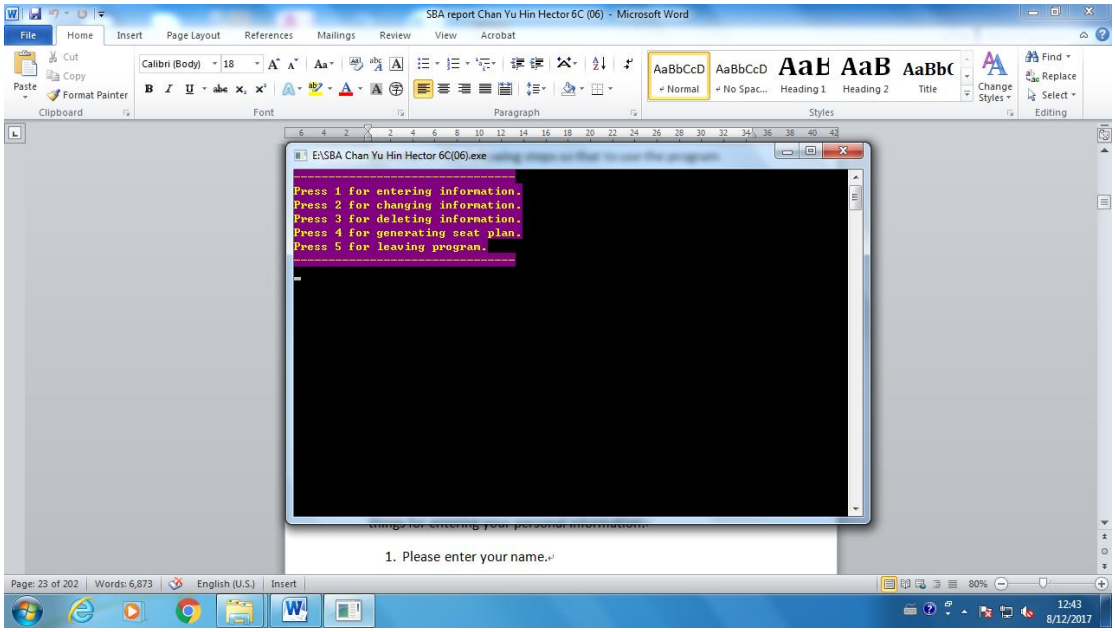
There are many steps for using the program. Please pay attention in the following steps so that to use the program properly.

1. Open the files.



2. Click the program called ‘Seating plan’

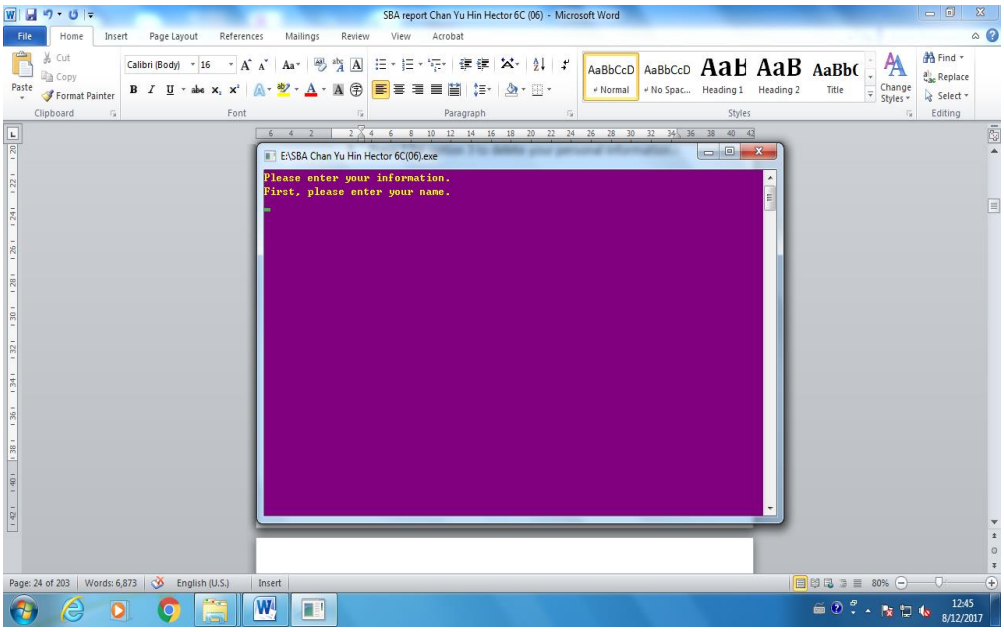
3. There are 5 options for you to choose:



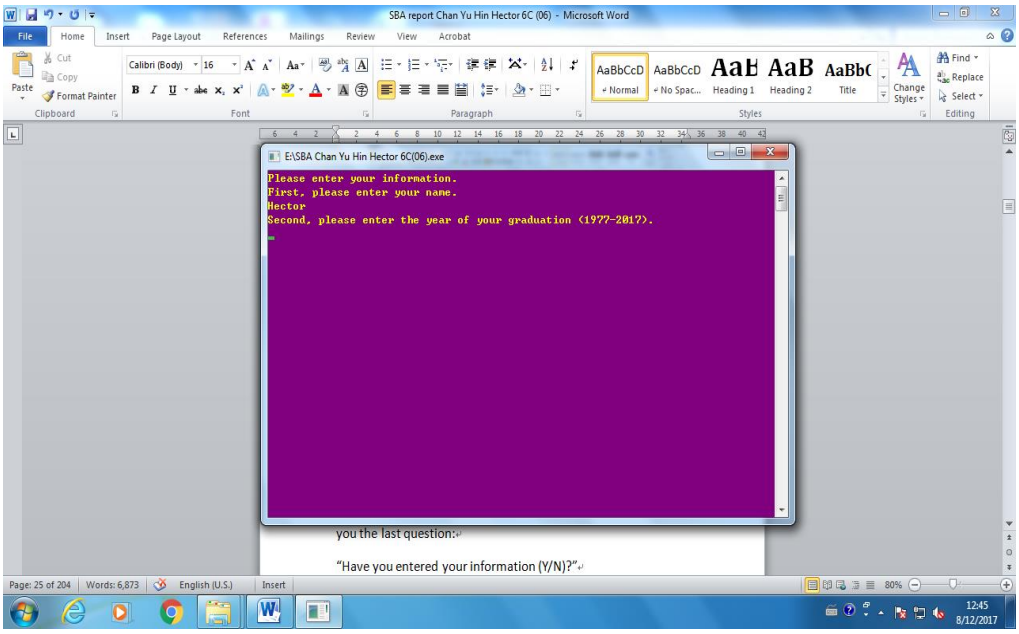
1. Press 1 for option 1 to enter your personal information.
2. Press 2 for option 2 to change your personal information.
3. Press 3 for option 3 to delete your personal information.
4. Press 4 for option 4 to generate the seating plan.
5. Press 5 for option 5 to exit the program.

4. If option 1 is chosen, the program will then ask you to do 5 things for entering your personal information:

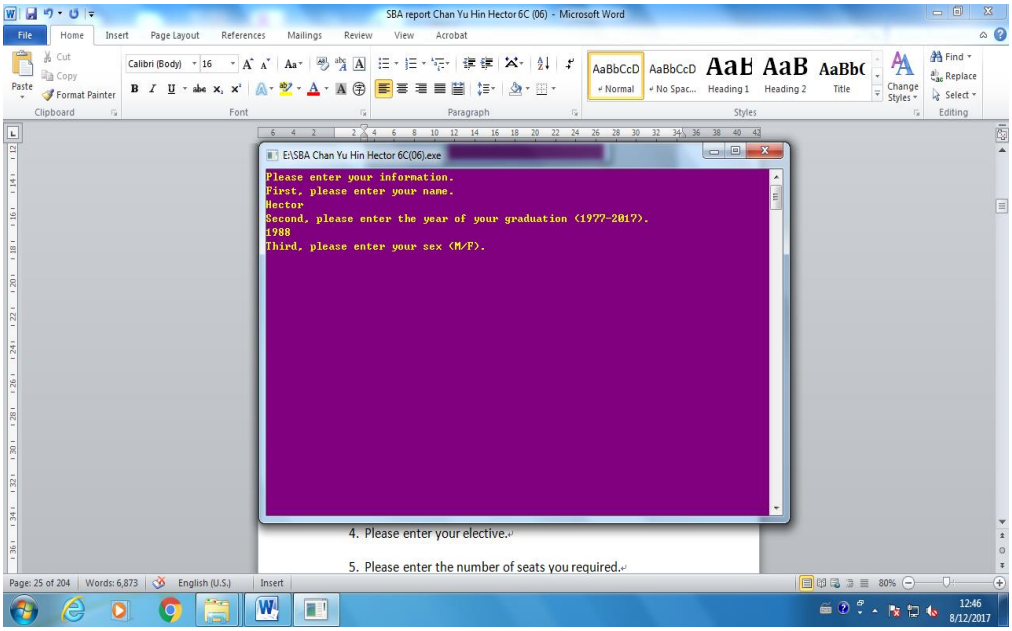
1. Please enter your name.



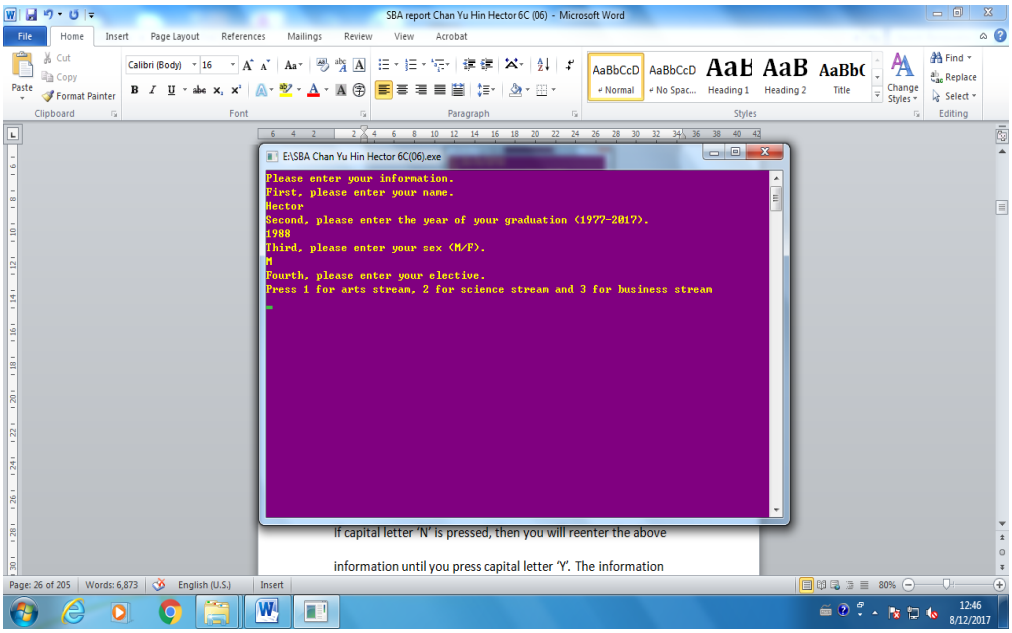
2. Please enter your year of the graduation.



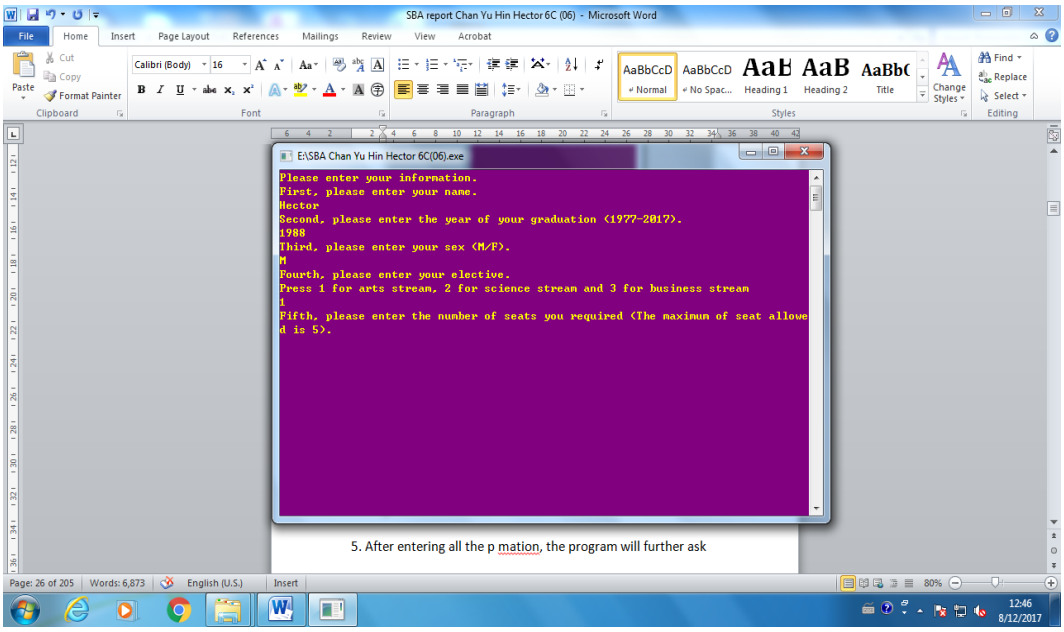
3. Please enter your sex.



4. Please enter your elective.

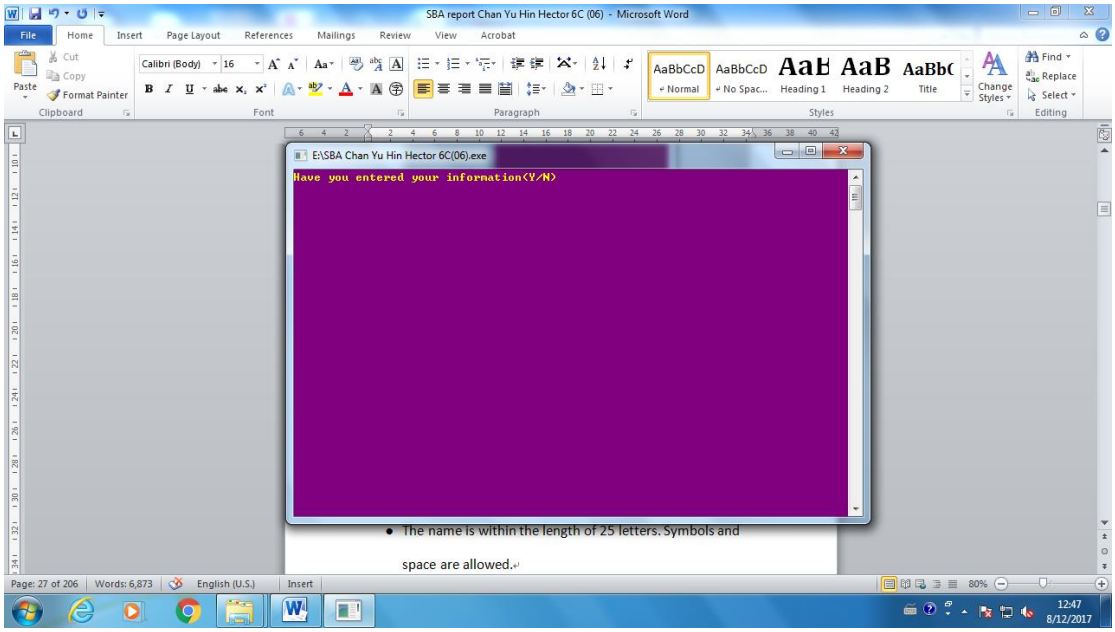


5. Please enter the number of seats you required.



5. After entering all the personal information, the program will further ask you the last question:

“Have you entered your information (Y/N)?”

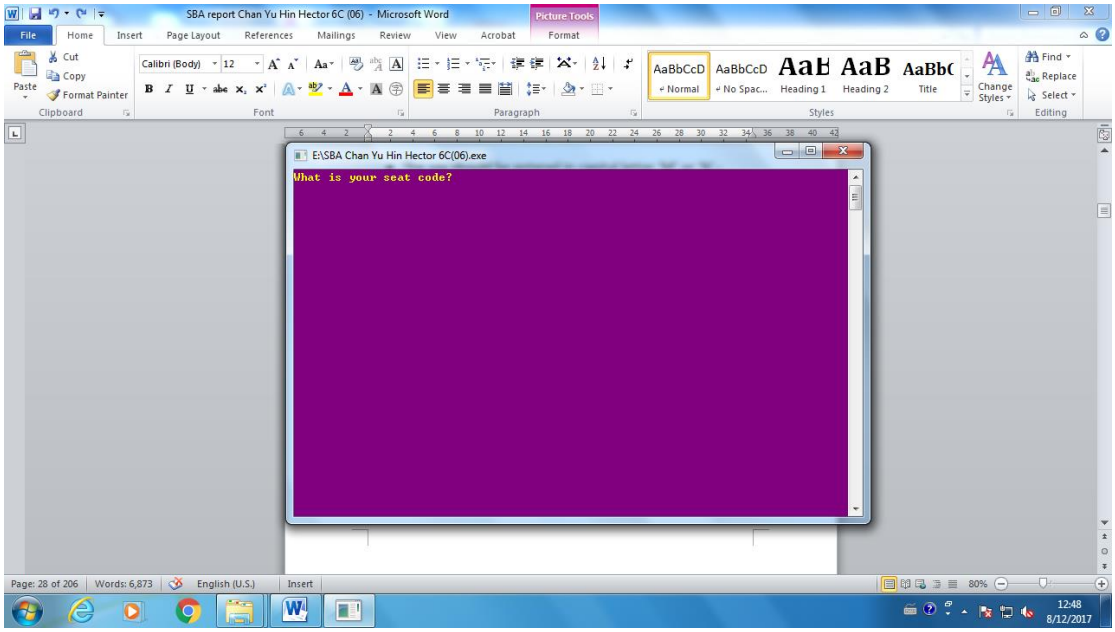


If capital letter ‘N’ is pressed, then you will reenter the above information until you press capital letter ‘Y’. The information then saved into the data file and return to the main menu.

6. There is also several reminding for entering information.

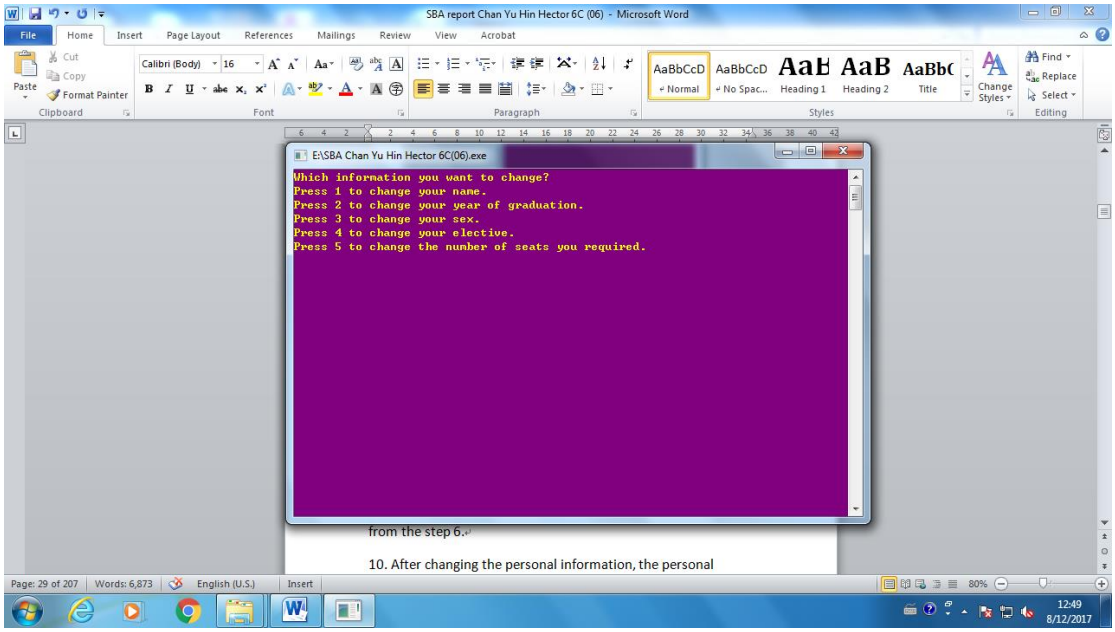
- The name is within the length of 25 letters. Symbols and space are allowed.
- The year of graduation should be entered within the range of 1977 to 2017.
- The sex should be entered in capital letter ‘M’ or ‘N’.
- The elective should be entered by 1, 2 and 3. There is error occupied if other type of data, number is entered
- The maximum of seat required is 5. If the number of seats required is out of the range, ‘Invalid number of seats required’ will be occupied.

6. If option 2 is chosen, the program will ask your seat code. The seat code is allocated according to the order of each participant entering its information.



8. After entering the seat code, the program will then give you 5 choices to choose again.

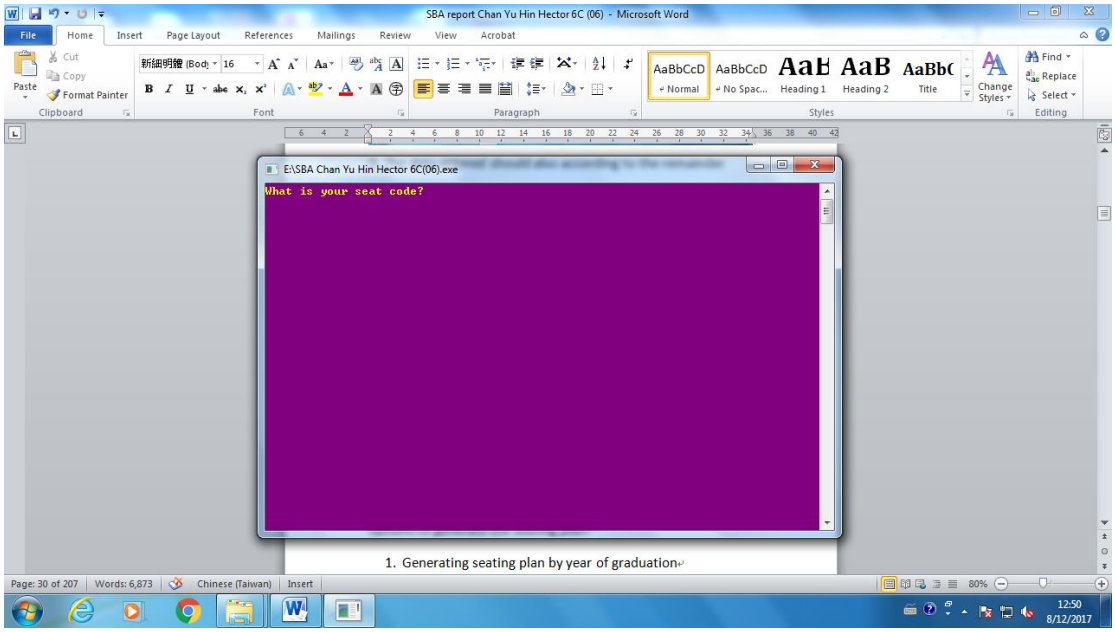
1. Change your name
2. Change your year of graduation
3. Change your sex
4. Change your elective
5. Change the number of seats you required.



9. The data entered should also according to the remainder from the step 6.

10. After changing the personal information, the personal information will be stored in the data file and return back to the main menu.

11. If option 3 is chosen, the program will ask your seat code as like as the step 7.

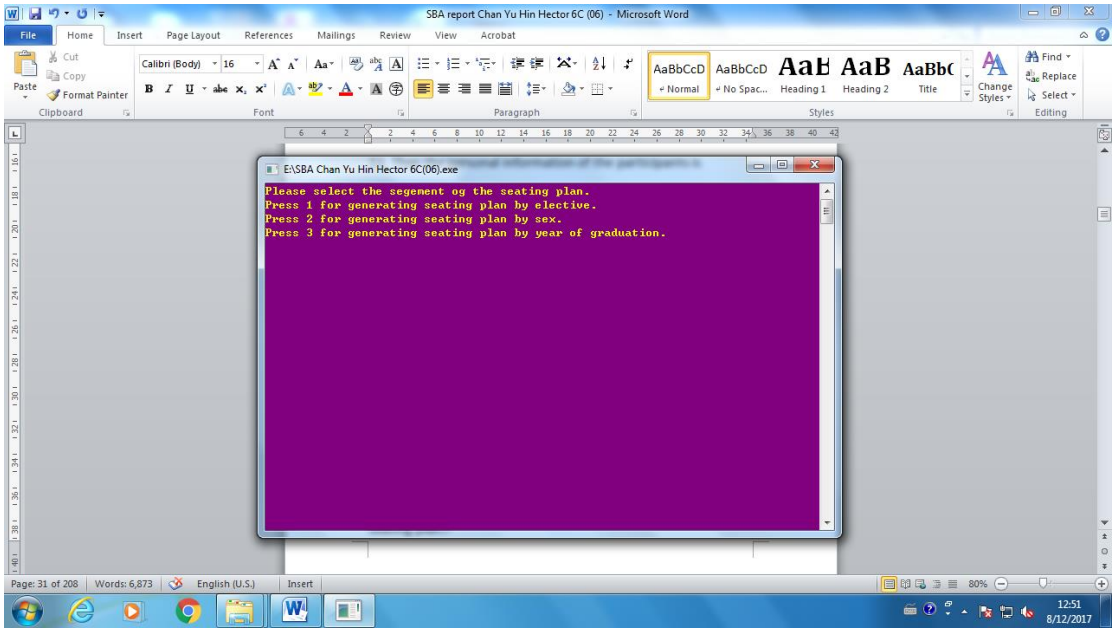


12. Please enter your seat code.

13. Then the personal information of the participants is deleted.

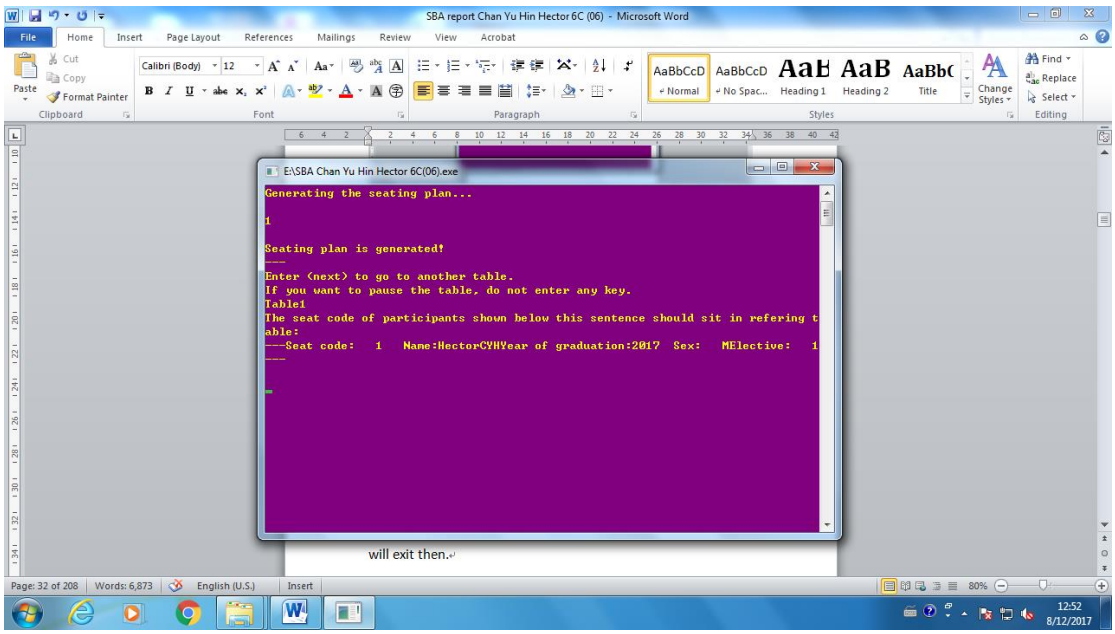
14. If option 4 is chosen, the program will then give you 3 options to generate the seating plan

1. Generating seating plan by year of graduation
2. Generating seating plan by sex
3. Generating seating plan by elective

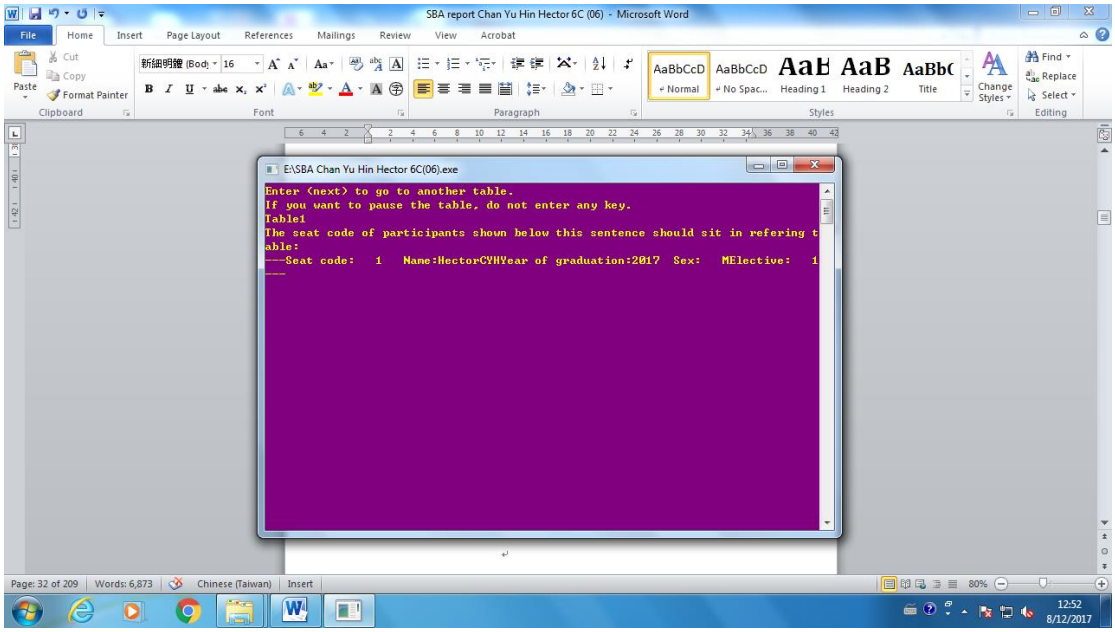


15. If one of the above choices is chosen, the program will then generate the seating plan.

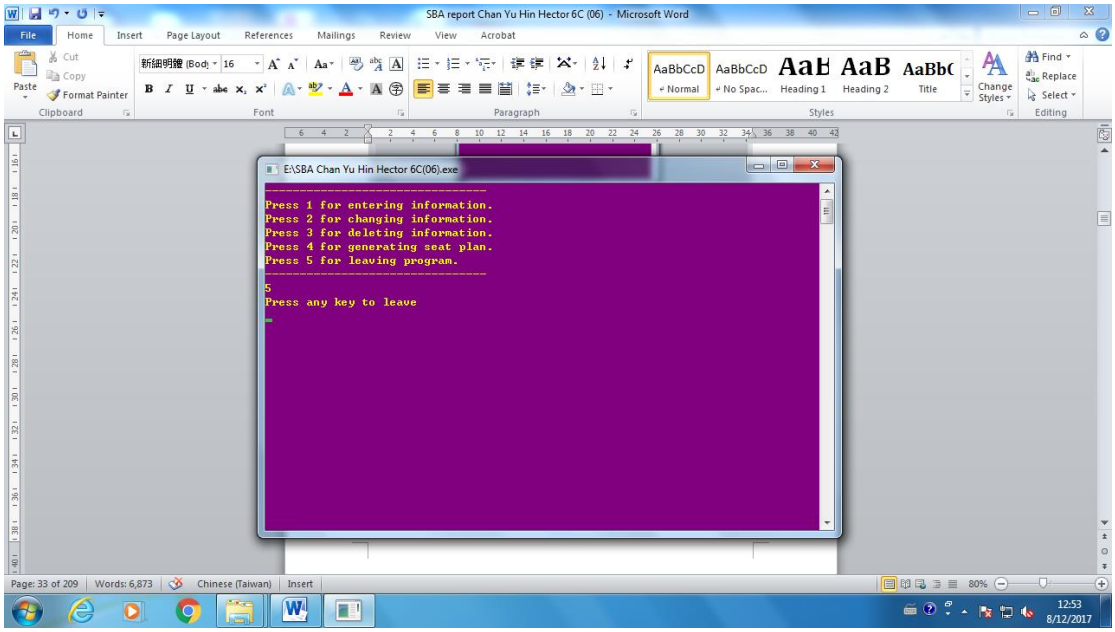
16. Please press enter to continue to show the display of the seating plan.



17. After the display of the seating plan, the seating plan will then save into the seating plan file. Please press ‘next’ to continue and back to the main menu.



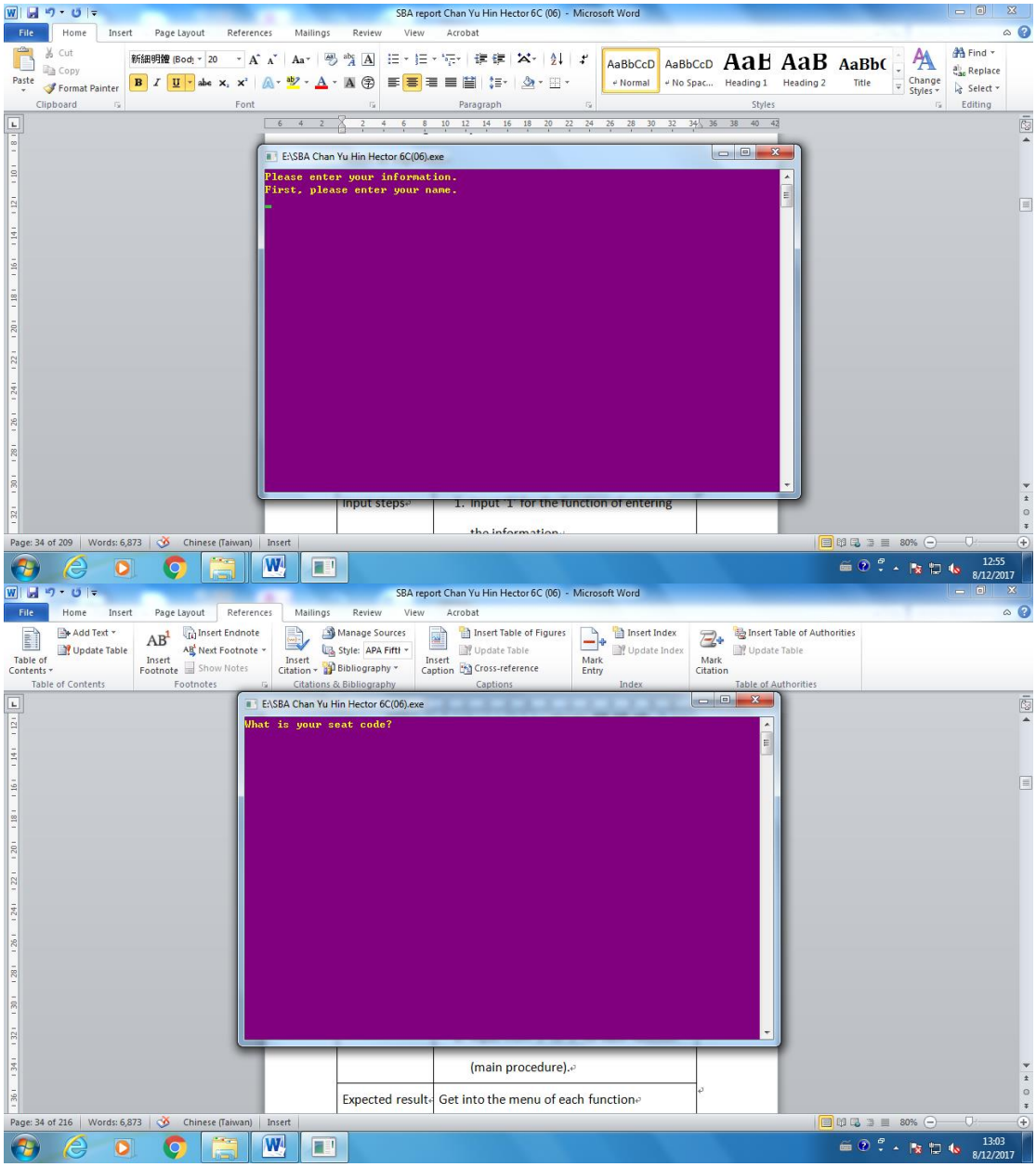
18. If option 5 is chosen, the program will then tell you to press any key to leave the program. If any key is pressed, the program will exit then.

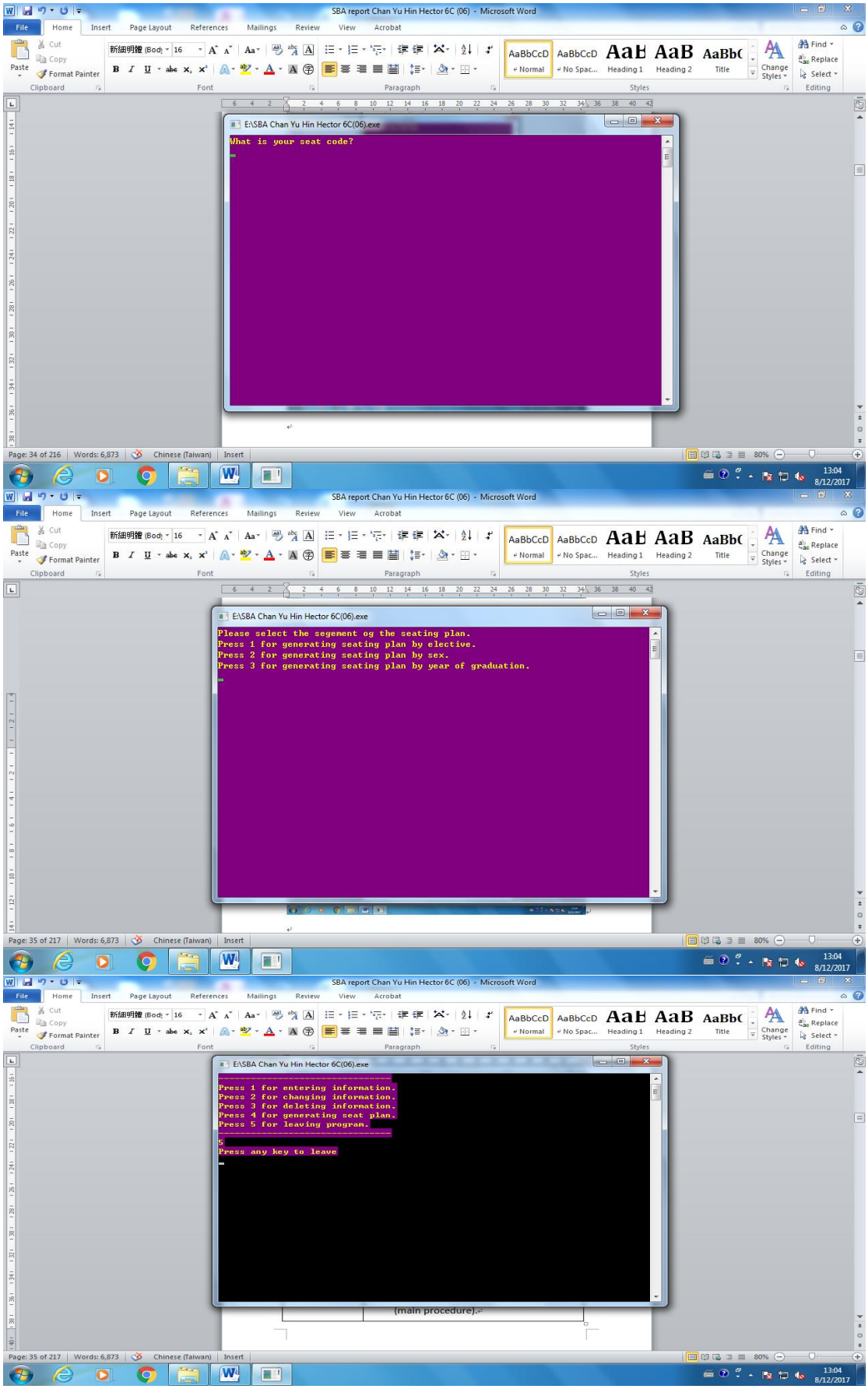


Testing and Evaluation

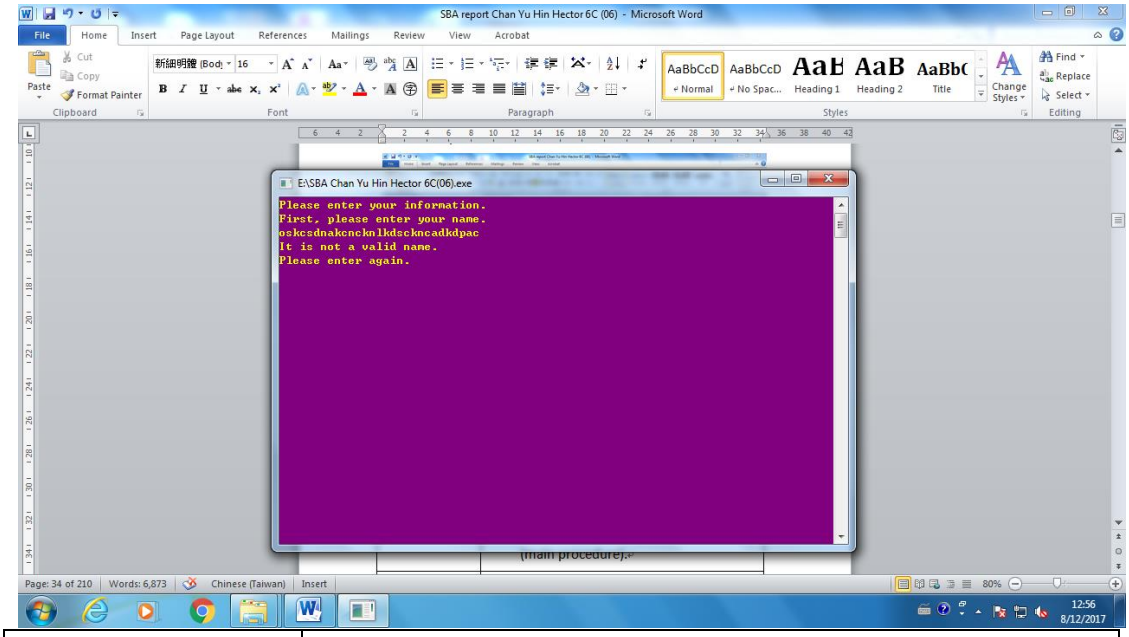
Testing

In the following, all major function will be tested and see if the result as expected.



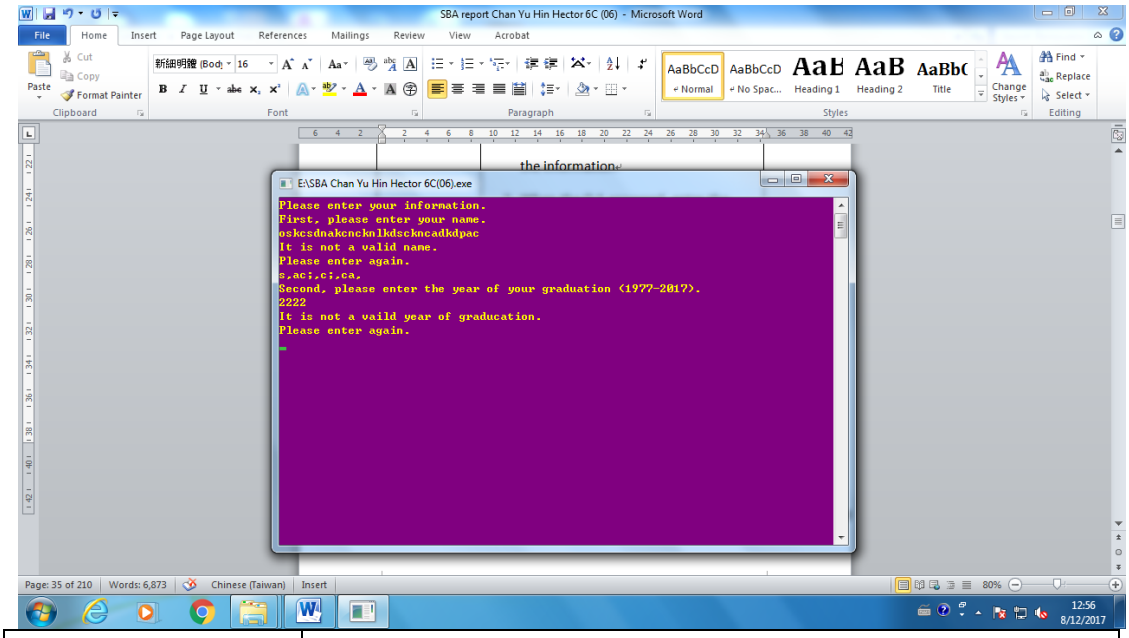

SBA report Chan Yu Hin Hector 6C (06) - Microsoft Word
E:\SBA Chan Yu Hin Hector 6C(06).exe
What is your seat code?
Page: 34 of 216
E:\SBA Chan Yu Hin Hector 6C(06).exe
Please select the segement og the seating plan.
Press 1 for generating seating plan by elective.
Press 2 for generating seating plan by sex.
Press 3 for generating seating plan by year of graduation.
Page: 35 of 217
E:\SBA Chan Yu Hin Hector 6C(06).exe
Press 1 for entering information.
Press 2 for changing information.
Press 3 for deleting information.
Press 4 for generating seat plan.
Press 5 for leaving program.
5
Press any key to leave
(main procedure).
Page: 35 of 217

Test case	Input a valid option in the main menu
Input steps	1. Under the main menu, the user is asked for option between ‘1’ to ‘5’ 2. Input from ‘1’ to ‘5’ for each function (main procedure).
Expected result	Get into the menu of each function
Actual result	Get into the menu of each function
Test case status	Passed



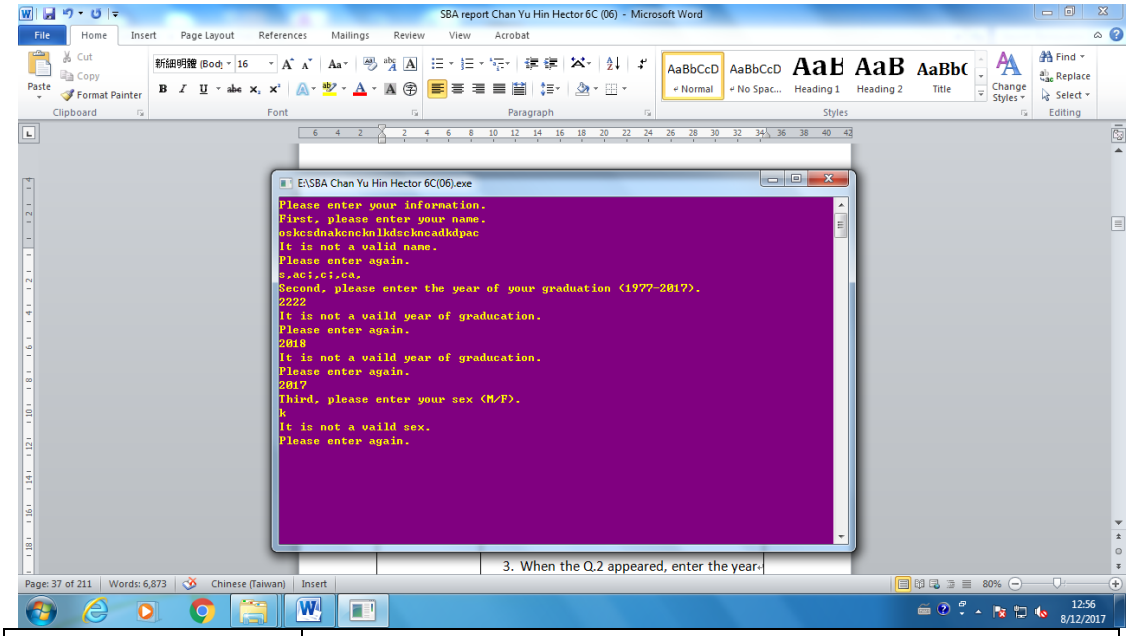
Test case	Check if an invalid name is not allowed
Input steps	1. Input ‘1’ for the function of entering the information

	2. When the Q.1 appeared, enter the name with over 25 lengths.
Expected result	‘It is not a valid name’ is appeared
Actual result	‘It is not a valid name’ is appeared
Test case status	Passed



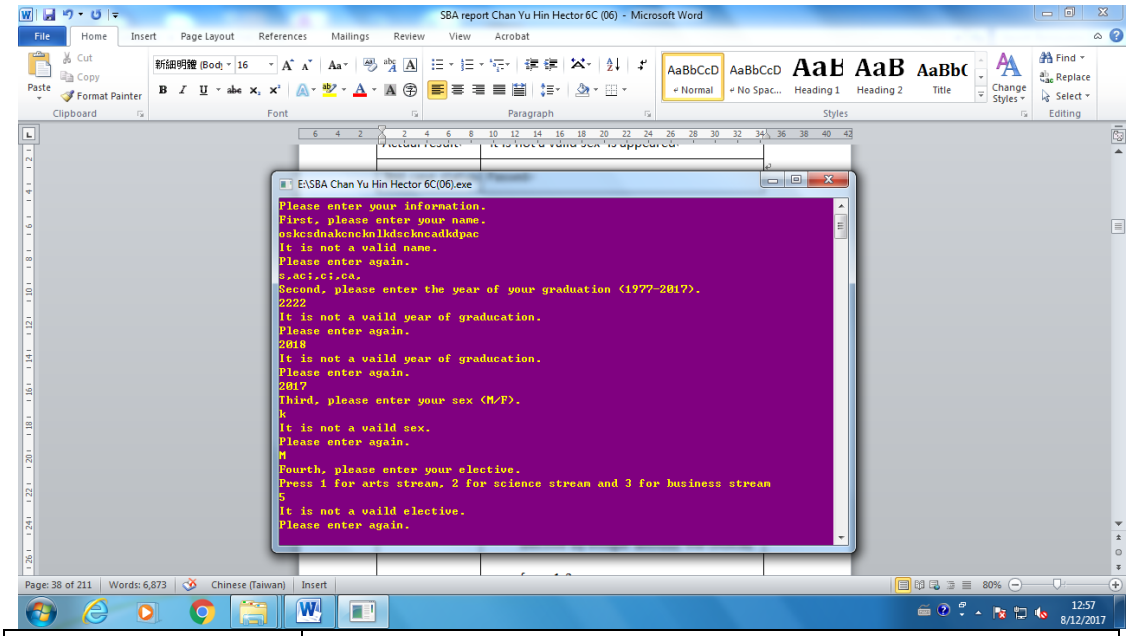
Test case	Check if the year of graduation is entered without the range provided is not allowed
Input steps	1. Input ‘1’ for the function of entering the information 2. When the Q.1 appeared, enter the name.

	3. When the Q.2 appeared, enter the year with the range out of the year of graduation.
Expected result	‘It is not a valid year of graduation’ is appeared
Actual result	‘It is not a valid year of graduation’ is appeared
Test case status	Passed



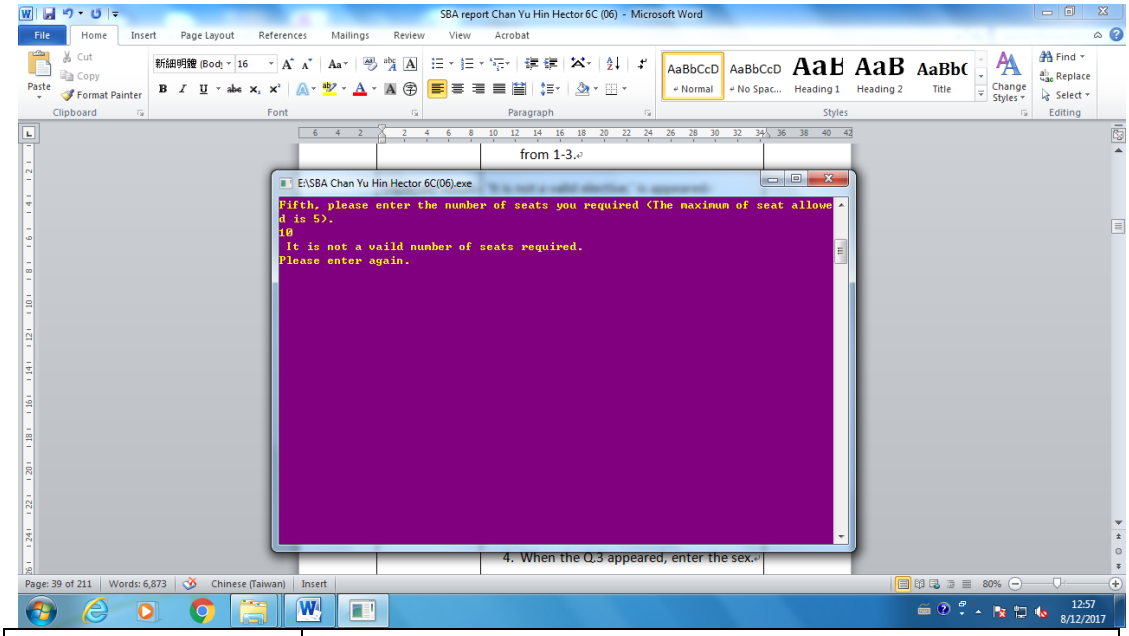
Test case	Check if the sex is entered without the field provided is not allowed
Input steps	1. Input ‘1’ for the function of entering

	the information 2. When the Q.1 appeared, enter the name. 3. When the Q.2 appeared, enter the year 4. When the Q.3 appeared, enter the sex with the character except ‘M’ and ‘F’.
Expected result	‘It is not a valid sex’ is appeared
Actual result	‘It is not a valid sex’ is appeared
Test case status	Passed



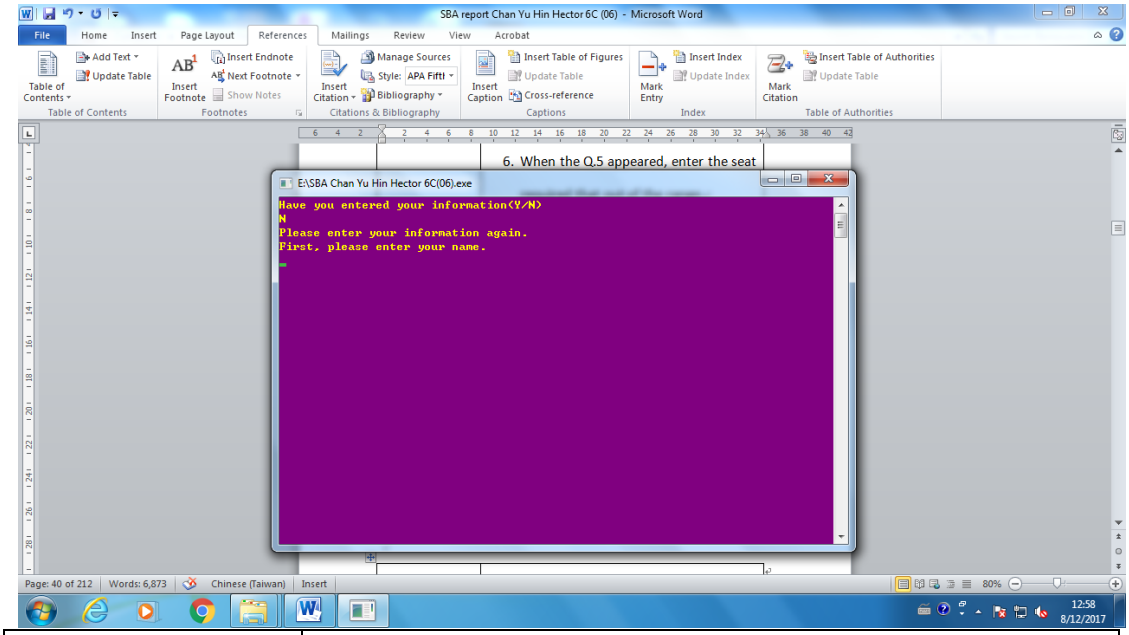
Test case	Check if the elective is entered without the choices provided is not allowed

Input steps	1. Input ‘1’ for the function of entering the information 2. When the Q.1 appeared, enter the name. 3. When the Q.2 appeared, enter the year 4. When the Q.3 appeared, enter the sex. 5. When the Q.4 appeared, enter the elective by integer without the choices from 1-3.
Expected result	‘It is not a valid elective.’ is appeared
Actual result	‘It is not a valid elective.’ is appeared
Test case status	Passed



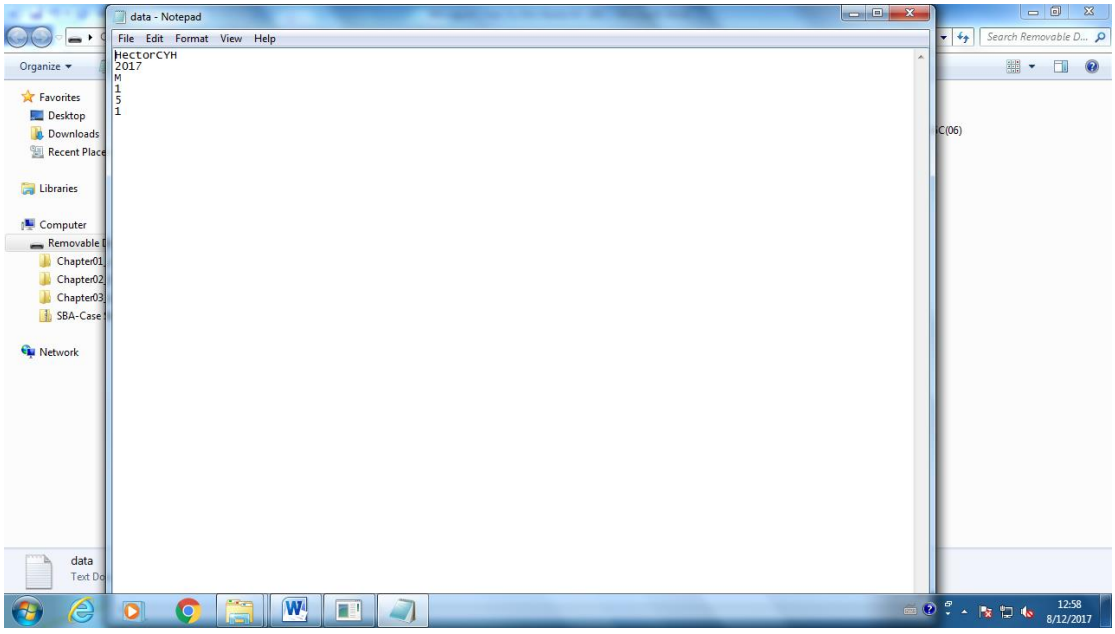
Test case	Check if the user's seat required is out of the maximum is not allowed.
Input steps	1. Input '1' for the function of entering the information 2. When the Q.1 appeared, enter the name. 3. When the Q.2 appeared, enter the year 4. When the Q.3 appeared, enter the sex. 5. When the Q.4 appeared, enter the elective. 6. When the Q.5 appeared, enter the seat required that out of the range.

Expected result	‘It is not a valid seat required.’ is appeared
Actual result	‘It is not a valid seat required.’ is appeared
Test case status	Passed

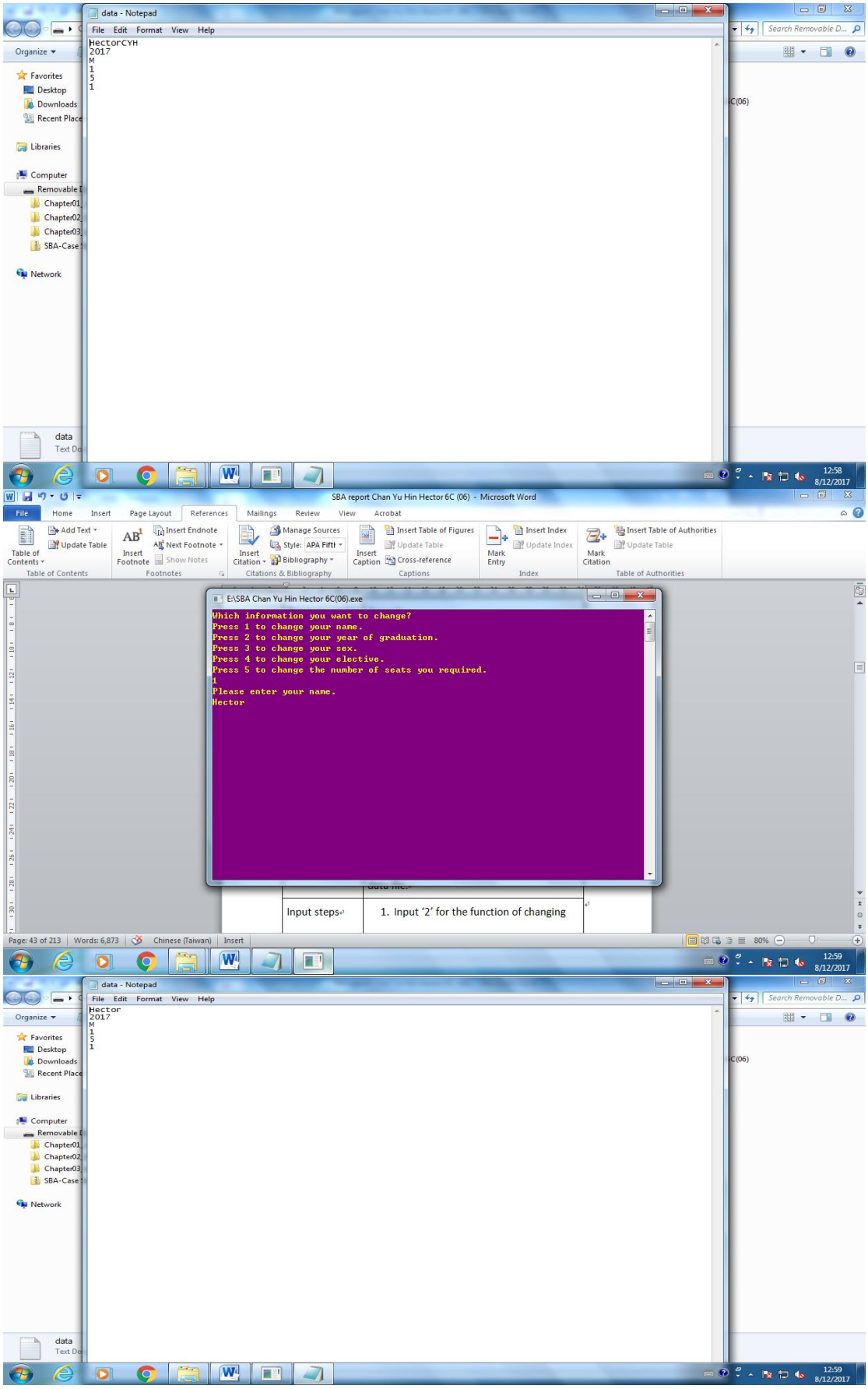


Test case	Check if the function of entering information can be used again
Input steps	1. Input ‘1’ for the function of entering the information 2. When the Q.1 appeared, enter the name. 3. When the Q.2 appeared, enter the year 4. When the Q.3 appeared, enter the sex.

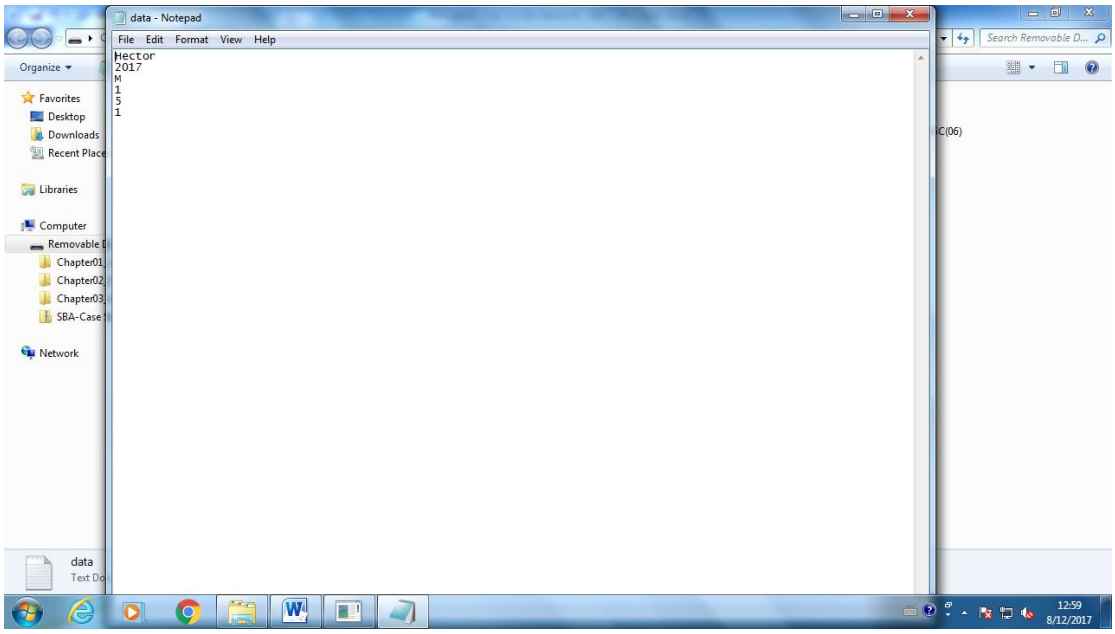
	5. When the Q.4 appeared, enter the elective. 6. When the Q.5 appeared, enter the seat required. 7. When the Q.6 appeared, press ‘N’. for entering the information again.
Expected result	Get back to the menu of function entering the information
Actual result	Get back to the menu of function entering the information
Test case status	Passed

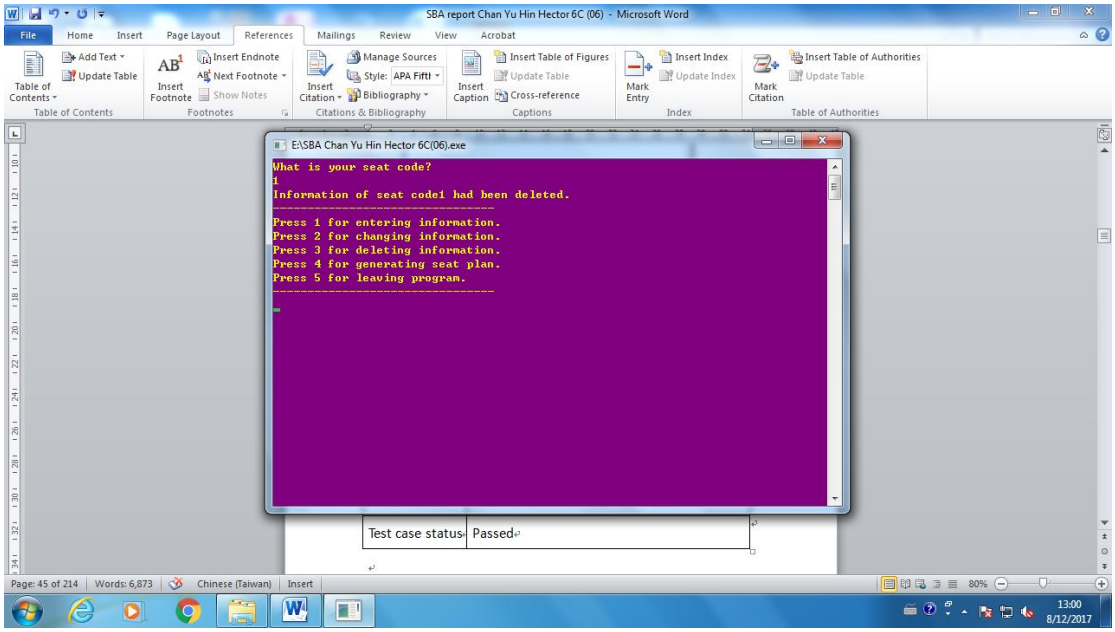


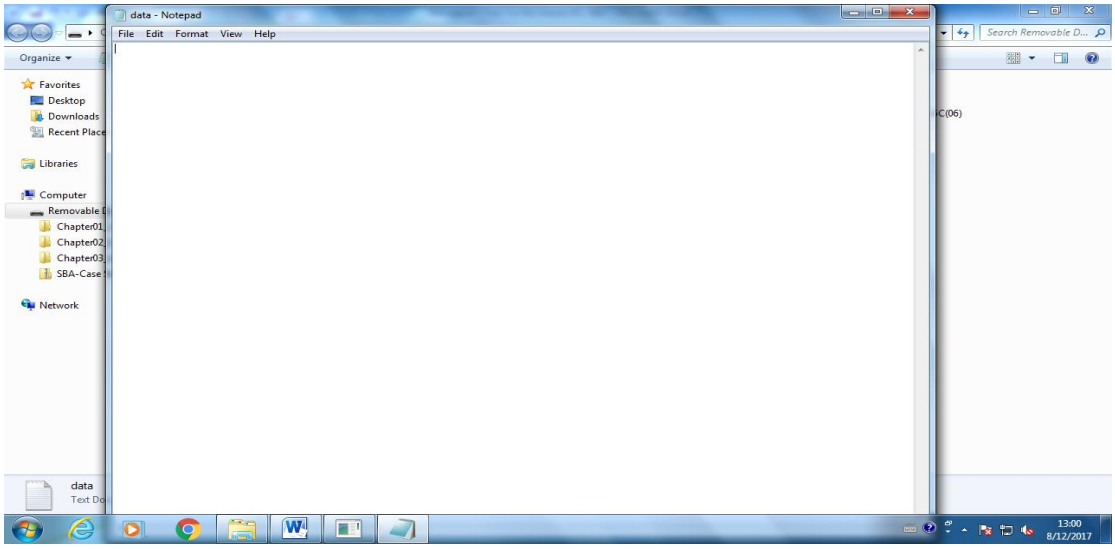
Test case	Check if the information is entered in the data file.
Input steps	1. Input ‘1’ for the function of entering the information 2. When the Q.1 appeared, enter the name. 3. When the Q.2 appeared, enter the year 4. When the Q.3 appeared, enter the sex. 5. When the Q.4 appeared, enter the elective. 6. When the Q.5 appeared, enter the seat required. 7. When the Q.6 appeared, press ‘Y’.
Expected result	Information is written in the data file.
Actual result	Information is written in the data file.
Test case status	Passed


data - Notepad
File Edit Format View Help
HectorCYH
2017
M
1
5
1
SBA report Chan Yu Hin Hector 6C (06) - Microsoft Word
E:\SBA Chan Yu Hin Hector 6C(06).exe
Which information you want to change?
Press 1 to change your name.
Press 2 to change your year of graduation.
Press 3 to change your sex.
Press 4 to change your elective.
Press 5 to change the number of seats you required.
1
Please enter your name.
Hector
Input steps
1. Input '2' for the function of changing
data - Notepad
File Edit Format View Help
Hector
2017
M
1
5
1

Test case	Check if the function of changing the information can change the information in data file.
Input steps	1. Input ‘2’ for the function of changing the information 2. Enter the seat code 3. From ‘1’ to ‘5’, choose each of them to change the information.
Expected result	Information in the data file is changed.
Actual result	Information in the data file is changed.
Test case status	Passed

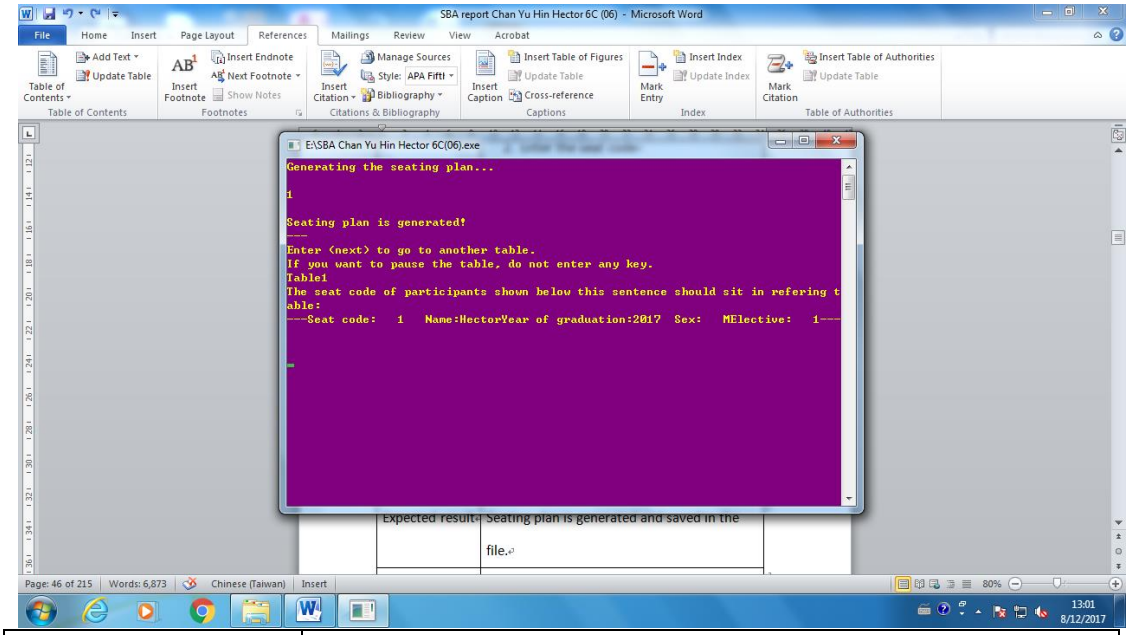






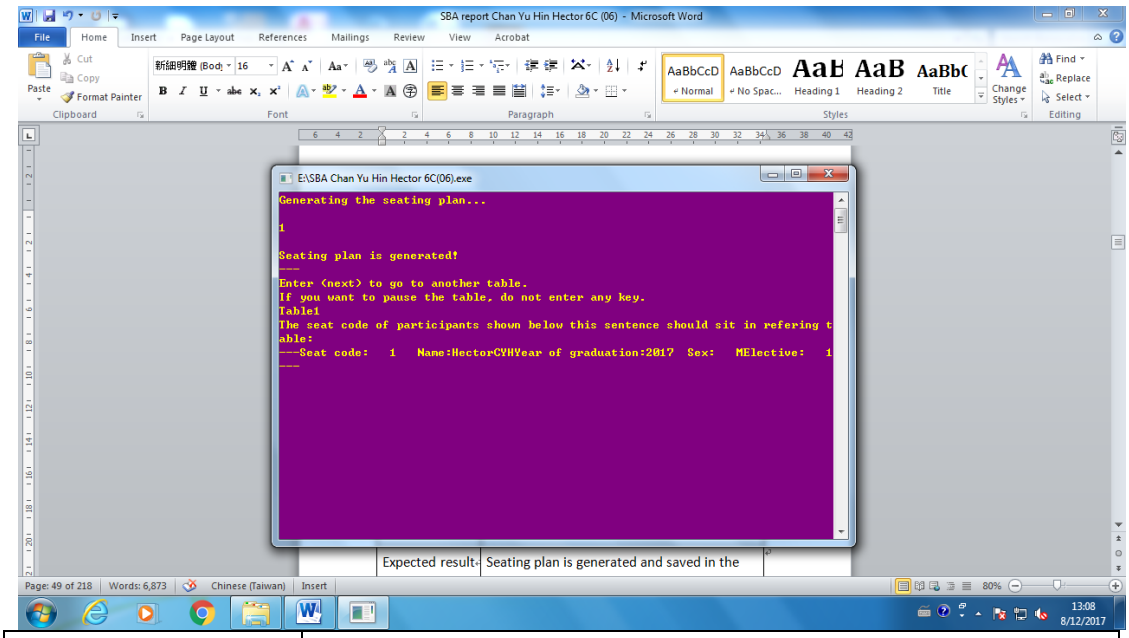
Test case	Check if the function of deleting the information can delete the information in data file.
Input steps	1. Input '3' for the function of deleting the information 2. Enter the seat code

Expected result	Information in the data file is deleted.
Actual result	Information in the data file is deleted.
Test case status	Passed



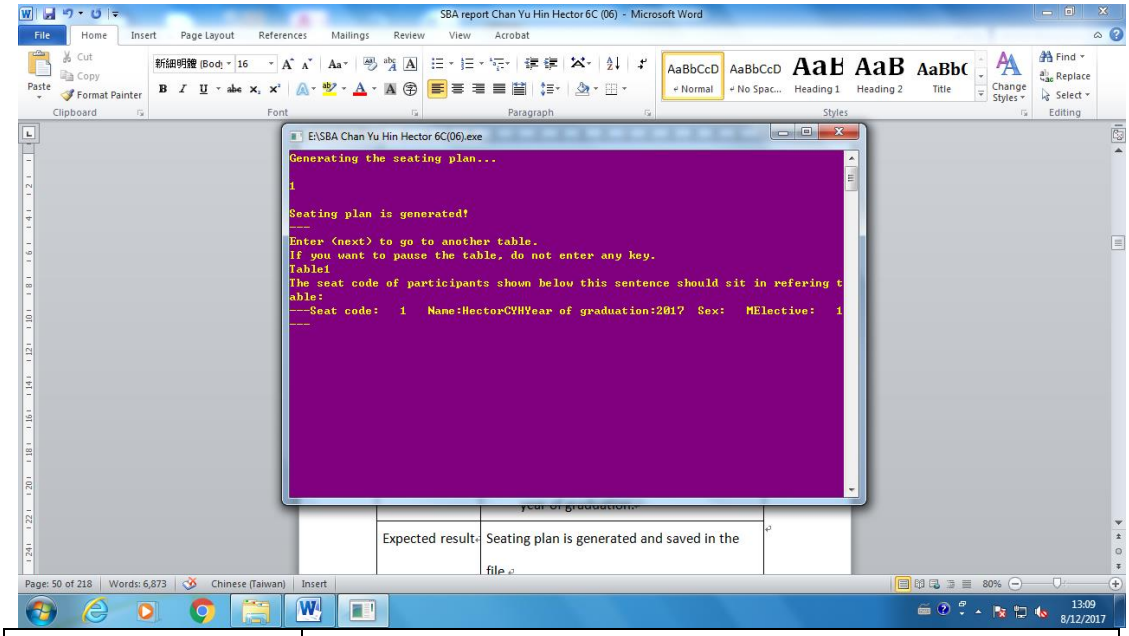
Test case	Check if the function of generating the seat plan by elective is successes.
Input steps	1. Input ‘4’ for the function of generating eating plan. 2. Press ‘1’ for generating seating plan by elective.
Expected result	Seating plan is generated and saved in the file.

Actual result	Seating plan is generated and saved in the file.
Test case status	Passed



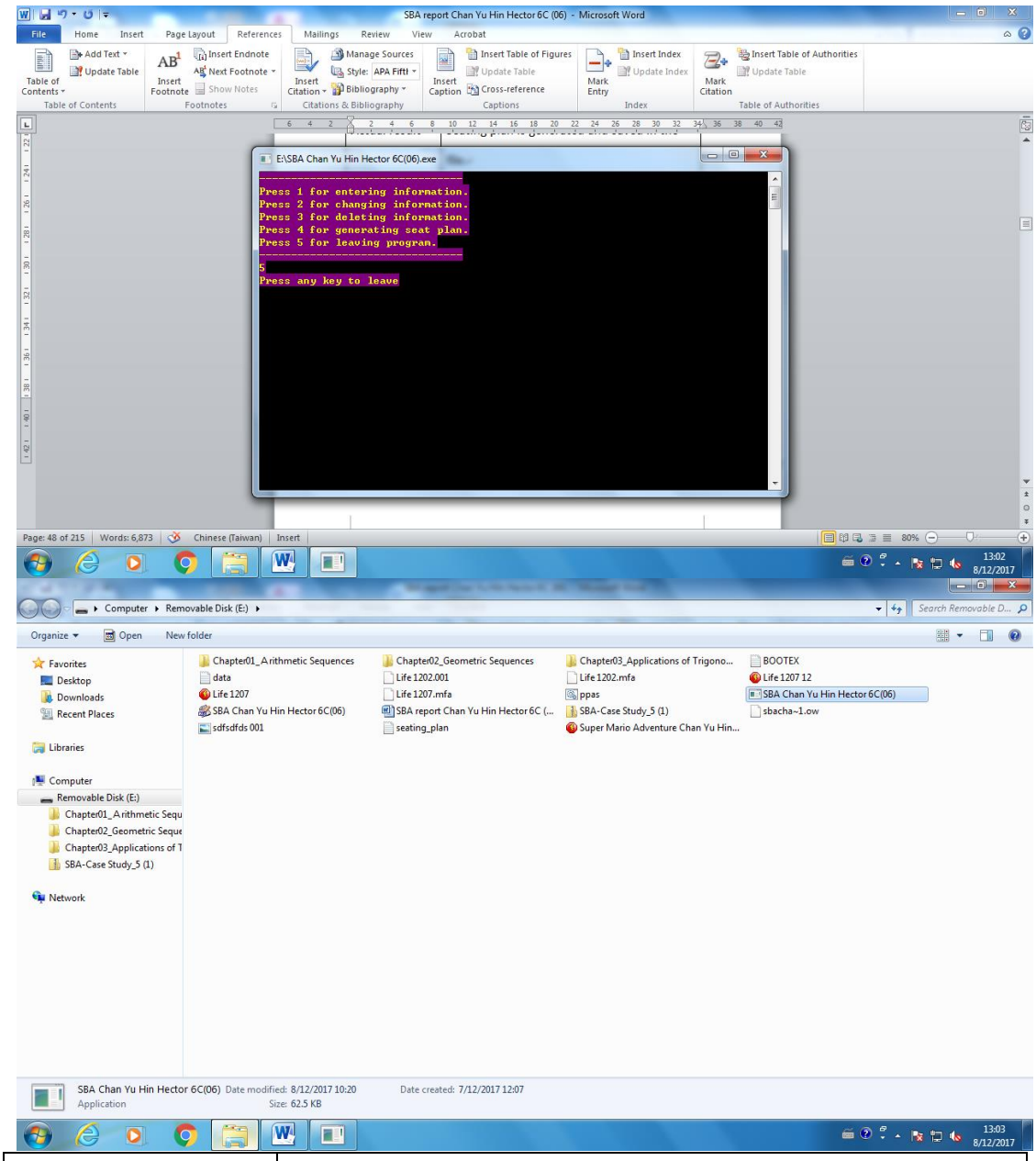
Test case	Check if the function of generating the seat plan by sex is successes.
Input steps	1. Input ‘4’ for the function of generating eating plan. 2. Press ‘2’ for generating seating plan by sex.
Expected result	Seating plan is generated and saved in the file.

Actual result	Seating plan is generated and saved in the file.
Test case status	Passed



Test case	Check if the function of generating the seat plan by year of graduation is successes.
Input steps	1. Input ‘4’ for the function of generating eating plan. 2. Press ‘1’ for generating seating plan by year of graduation.
Expected result	Seating plan is generated and saved in the file.

Actual result	Seating plan is generated and saved in the file.
Test case status	Passed



Test case	Check if the function of exit the program is success

Input steps	1. Input ‘5’ for the function of exiting the program 2. Press any button for leaving the program
Expected result	Success to leave the program.
Actual result	Success to leave the program.
Test case status	Passed

Evaluating

In this program, there is several strength and weakness as the followings:

Strength: Compared to other’s work, this program has provided more choice for the participants to enter their personal information and different types of generating seating plans, such as by elective or year of graduation. Also, this program has set up to face different condition. For example, this participant in group sits or not, no participants sit on this table or the seats per table is less than 10.

Weakness: First, there is too much global value in the program. There should be improvement for the segment of the global and local value. Second, there is too long operation in the program. There should be some way out to simplify the program's operation. Third, there is no seat code shown in the program. It should be added after the input of information. Fourth, the display of the program is Command Line Interface (CLI), which would be so hard for the user with the digital-divide to use.

Conclusion

My work has fulfilled the objectives of the coursework. It aims to develop a program for the dinner registration and generating a seating plan. My works has fulfilled the objectives and can generate seating plan with different method such as by elective, year of graduation or sex.

Skills Obtained

From this work, I have learned several new function of the Pascal, such as the ‘append’. I also learned how to use procedure to solve each logical questions, how to make the program become user-friendly and also how to set a better data structure.

Difficulties Encountered

The most difficult part in building the program is to think, analysis how to group participants with same data and sort, divide them into different group and further into the table. To resolve these problems, I have set up two procedures. One for grouping of the participants with the same data. One for sorting and dividing the participants into the table by sorted data from the previous procedure. So that the logical problems can be solved individually and clearly. However, there are still many bugs, error appeared in the execution. Therefore, I have to use different types of data or writing something in the specific line of the program. So that to solve some carless, minor mistakes, errors in the program, which takes longer time to resolve all the

above difficulties

.

Future Development

Assuming that I have more time and resources available, I will propose some feasible improvement. First, I will rectify the problem of length of the program. It is too long for the program and it is believed that it can be improved, replaces with a shorter statement or by procedure. Second, the display of the user-interface and the seating plan can be improved. It is believed that it can be changed by removing their current place in the Pascal with changing the command.

Reference & Acknowledgement

1. SBA-Case Study_5
2. Longman- New senior Secondary Information and Communication Technology Elective D1
3. Longman- New senior Secondary Information and Communication Technology Elective D2
4. Free Pascal
   Website:
   https://www.freepascal.org/docs-html/rtl/system/index-8.html

Appendix

Program code

```
Program SBA(seatingplan);
uses Crt;
const
secondmax=500;
var
data,d:text;
name,answer,i:string;
sex : char;
elective,seat,year,a,b,numberofpeople,quarry,counter,e,
choice,top:integer;
stay,deletecode,f,g,h,group1number,group2number,grou
p3number,group4number,group5number,group6number
,group7number,group8number,group9number,space:int
eger;
group1sit,group2sit,group3sit,group4sit,group5sit,group
6sit,group7sit,group8sit,group9sit,hold,final,numberofta
```

```
ble,code:integer;
finaltable,numberofparticipant,require,tablenumber:inte
ger;
table,j,k,l,m,n,o:integer;
found,check,group1finish,group2finish,group3finish,gro
up4finish,group5finish,group6finish,group7finish,group8
finish,group9finish,quarryfound:boolean;
loaddata:array[1..secondmax] of integer;
readname:array[1..secondmax] of string[25];
readyear:array[1..secondmax] of integer;
readsex:array[1..secondmax] of char;
readelective:array[1..secondmax] of integer;
readseat:array[1..secondmax] of integer;
readseatcode:array[1..secondmax] of integer;
group1data,group2data,group3data,group4data,group5
data,group6data,group7data,group8data,group9data:arr
ay[1..secondmax] of integer;
finalnumber:array[1..secondmax] of integer;
tabledata:array[1..secondmax,1..10]of integer;
```

```
tablename:array[1..secondmax,1..10]of string[25];
tablesex:array[1..secondmax,1..10]of char;
tableelective:array[1..secondmax,1..10]of integer;
tableyear:array[1..secondmax,1..10]of integer;
fgroup,ggroup,hgroup,jgroup,kgroup,lgroup,mgroup,ngr
oup,ogroup:array[1..200] of boolean;
tablefull:array[1..secondmax] of boolean;
groupdata:array[1..secondmax] of integer;
loaddatause:array[1..secondmax] of boolean;

Procedure Name_Check;
begin
while (name=' ') or (length(name)>25) do
begin
writeln('It is not a valid name.');
writeln('Please enter again.');
readln(name);
end;
end;
```

```
Procedure YearOfGraducation_Check;
begin
while (year>2017) or (year<1977) do
begin
writeln('It is not a vaild year of graducation.');
writeln('Please enter again.');
readln(year);
end;
end;

Procedure Sex_Check;
begin
while (sex <> 'M') and (sex <> 'F') do
begin
writeln('It is not a vaild sex.');
writeln('Please enter again.');
readln(sex);
end;
```

```
end;

Procedure Elective_Check;
begin
while (elective<>1) and (elective<>2) and (elective<>3)
do
begin
writeln('It is not a vaild elective.');
writeln('Please enter again.');
readln(elective);
end;
end;

Procedure NumberOfSeats_Check;
begin
while (seat>5) or (seat<0) do
begin
writeln(' It is not a vaild number of seats required.');
writeln('Please enter again.');
```

```
readln(seat);
end;
end;

Procedure EnterAgain;
begin
writeln('Have you entered your information(Y/N)');
readln(answer);
if answer='N' then
begin
writeln('Please enter your information again.');
writeln('First, please enter your name.');
readln(name);
Name_Check;
writeln('Second, please enter the year of your
graduation (1977-2017).');
readln(year);
YearOfGraduation_Check;
writeln('Third, please enter your sex(M/F).');
```

```
readln(sex);
Sex_Check;
writeln('Fourth, please enter your elective.');
writeln('Press 1 for arts stream, 2 for science stream and
3 for business stream');
readln(elective);
Elective_Check;
writeln('Fifth, please enter the number of seats you
required (The maximum of seat allowed is 5).');
readln(seat);
NumberOfSeats_Check;
end;
end;

Procedure WriteInformation;
   begin
      assign(data,'data.txt');
      append(data);
      a:=loaddata[top];
```

```
        writeln(data,name);
        writeln(data,year);
        writeln(data,sex);
        writeln(data,elective);
        writeln(data,seat);
        writeln(data,a+1);
        a:=a+1;
        close(data);
    end;

Procedure ReadInformation;
    begin
        assign(data,'data.txt');
        reset(data);
        b:=1;
        numberofpeople:=0;
        while not eof(data) do
            begin
                readln(data,readname[b]);
```

```
            readln(data,readyear[b]);
            readln(data,readsex[b]);
            readln(data,readelective[b]);
            readln(data,readseat[b]);
            readln(data,loaddata[b]);

numberofpeople:=numberofpeople+readseat[b];
            top:=b;
            b:=b+1;
        end;
        close(data);
    end;

Procedure EnterInformation;
  begin
    clrscr;
    writeln('Please enter your information.');
    writeln('First, please enter your name.');
    readln(name);
```

```
        Name_Check;
        writeln('Second, please enter the year of your
graduation (1977-2017).');
        readln(year);
        YearOfGraducation_Check;
        writeln('Third, please enter your sex (M/F).');
        readln(sex);
        Sex_Check;
        writeln('Fourth, please enter your elective.');
        writeln('Press 1 for arts stream, 2 for science
stream and 3 for business stream');
        readln(elective);
        Elective_Check;
        writeln('Fifth, please enter the number of seats you
required (The maximum of seat allowed is 5).');
        readln(seat);
        NumberOfSeats_Check;
        clrscr;
        EnterAgain;
```

```
        WriteInformation;
        ReadInformation;
    end;

Procedure ChangeInformation;
    begin
        clrscr;
        assign(data,'data.txt');
        b:=1;
        reset(data);
        while not eof (data) do
        begin
            readln(data,readname[b]);
            readln(data,readyear[b]);
            readln(data,readsex[b]);
            readln(data,readelective[b]);
            readln(data,readseat[b]);
            readln(data,readseatcode[b]);
            top:=b;
```

```
      b:=b+1;
end;
close(data);
writeln('What is your seat code?');
readln(quarry);
clrscr;
counter:=1;
found:=false;

while(counter<=top) and (found=false) do
   begin
   if quarry=readseatcode[counter]
   then begin
                found:=true;
                stay:=counter;
           end
   else
      found:=false;
      counter:=counter+1;
```

```
            end;
        if found=true then
        begin
            writeln('Which information you want to
change?');
            writeln('Press 1 to change your name.');
            writeln('Press 2 to change your year of
graduation.');
            writeln('Press 3 to change your sex.');
            writeln('Press 4 to change your elective.');
            writeln('Press 5 to change the number of seats
you required.');
            readln(a);
            case a of
            1:begin
                writeln('Please enter your name.');
                readln(name);
                Name_Check;
                readname[stay]:=name;
```

```
            end;
        2:begin
            writeln('Please enter your year of
graduation.');
            readln(year);
            YearOfGraducation_Check;
            readyear[stay]:=year;
          end;
        3:begin
            writeln('Please enter your sex.');
            readln(sex);
            Sex_Check;
            readsex[stay]:=sex;
          end;
        4:begin
              writeln('Please enter your elective.');
              writeln('Press 1 for art stream, 2 for
science stream and 3 for business stream.');
              readln(elective);
```

```
                    Elective_Check;
                    readelective[stay]:=elective;
                end;
            5:begin
                    writeln('Please enter the number of
seats you required.');
                    readln(seat);
                    NumberOfSeats_Check;
                    readseat[stay]:=seat;
                end;
            end;
        end;
    if found=false
    then writeln('This seat code is not existed.')
    else if found = true
    then begin
            assign(d,'data.txt');
            rewrite(d);
            writeln(readname[counter]);
```

```
                e:=1;
                repeat
                   writeln(d,readname[e]);
                   writeln(d,readyear[e]);
                   writeln(d,readsex[e]);
                   writeln(d,readelective[e]);
                   writeln(d,readseat[e]);
                   writeln(d,readseatcode[e]);
                   e:=e+1;
                      until e=(top+1);
                      close(d);
                      end;
                end;

Procedure DeleteInformation;
   begin
      clrscr;
      ReadInformation;
      writeln('What is your seat code?');
```

```
        readln(code);
        b:=1;
        check:=false;
        repeat
           if loaddata[b]=code
           then check:=true
           else b:=b+1;
           until(check = true) or (b=top);
           if check=true
           then begin
                      for deletecode:=b to (top) do
                           begin

readname[deletecode]:=readname[deletecode+1];

readsex[deletecode]:=readsex[deletecode+1];

readyear[deletecode]:=readyear[deletecode+1];
```

```
readelective[deletecode]:=readelective[deletecode+1];

readseat[deletecode]:=readseat[deletecode+1];

loaddata[deletecode]:=loaddata[deletecode+1];
                    end;
                writeln('Information of seat code',
code,' had been deleted.');
                end
            else
                writeln('Invalid input.');
                assign(d,'data.txt');
                rewrite(d);
                for b:=1 to (top-1) do
                begin
                    writeln(d, readname[b]);
                    writeln(d, readyear[b]);
                    writeln(d, readsex[b]);
                    writeln(d, readelective[b]);
```

```
                writeln(d, readseat[b]);
                writeln(d, loaddata[b]);
            end;
            close(d);
        end;


Procedure GroupingSameElective;
begin
    ReadInformation;
    f:=1;
    g:=1;
    h:=1;
    hold:=b;
    writeln;
    for b:=1 to hold do
        begin
            if readelective[b]=1
            then begin
```

```
                group1data[f]:= loaddata[b];
                f:=f+1;
              end
        else if readelective[b]=2
        then begin
                group2data[g]:=loaddata[b];
                g:=g+1;
                end
        else if readelective[b]=3
        then begin
                group3data[h]:=loaddata[b];
                h:=h+1;
                end
      end;
   group1number:=f;
   group2number:=g;
   group3number:=h;
end;
```

```
Procedure GroupingSameSex;
begin
   ReadInformation;
   k:=1;
   j:=1;
   hold:=b;
   writeln;
   for b:=1 to hold do
      begin
         if readsex[b]='M'
         then begin
                  group4data[k]:= loaddata[b];
                  k:=k+1;
                end
         else if readsex[b]='F'
         then begin
                  group5data[j]:=loaddata[b];
                  j:=j+1;
                  end
```

```
        end;
    group4number:=k;
    group5number:=j;
 end;

Procedure GroupingSameYearOfGraduation;
begin
   ReadInformation;
   l:=1;
   m:=1;
   n:=1;
   o:=1;
   hold:=b;
   writeln;
   for b:=1 to hold do
      begin
         if (readyear[b]>1976) and (readyear[b]<1988)
         then begin
                  group6data[l]:= loaddata[b];
```

```
                l:=l+1;
              end
        else if (readyear[b]>1987) and
(readyear[b]<1998)
        then begin
                group7data[m]:=loaddata[b];
                m:=m+1;
                end
        else if (readyear[b]>1997) and
(readyear[b]<2008)
        then begin
                group8data[n]:=loaddata[b];
                n:=n+1;
                end
        else if (readyear[b]>2007) and
(readyear[b]<2018)
        then begin
                group9data[o]:=loaddata[b];
                o:=o+1;
```

```
                    end
        end;
    group6number:=l;
    group7number:=m;
    group8number:=n;
    group9number:=o;
 end;


Procedure Table_Seat;
   begin
      if numberofpeople<=10
      then begin
                space:= numberofpeople;
                table:=1;
             end
       else begin
                space:=10;
                table:=numberofpeople div space+30;
```

```
                end;
    end;

Procedure GroupSit_Check;
  begin
    if group1sit=group1number
    then group1finish:=true;
    if group2sit=group2number
    then group2finish:=true;
    if group3sit=group3number
    then group3finish:=true;
  end;

Procedure GroupSit_Check2;
  begin
    if group4sit=group4number
    then group4finish:=true;
    if group5sit=group5number
    then group5finish:=true;
```

```
    end;

Procedure GroupSit_Check3;
    begin
        if group6sit=group6number
        then group6finish:=true;
        if group7sit=group7number
        then group7finish:=true;
        if group8sit=group8number
        then group8finish:=true;
        if group9sit=group9number
        then group9finish:=true;
    end;

Procedure OutputSeatingPlan;
    begin
        final:=1;
        for numberoftable:=1 to (finaltable) do
        begin
```

```
        repeat
            writeln('Enter (next) to go to another table.');
            writeln('If you want to pause the table, do not
enter any key.');
            writeln('Table',numberoftable);
            writeln('The seat code of participants shown
below this sentence should sit in refering table:');
            for numberofparticipant:=1 to
(finalnumber[final]) do
            begin
                writeln('---Seat code:',
tabledata[numberoftable,numberofparticipant]:4,'Name
:':8,tablename[numberoftable,numberofparticipant]:4,'Y
ear of graduation:':4,
tableyear[numberoftable,numberofparticipant]:4,
'Sex:':6,tablesex[numberoftable,numberofparticipant]:4,
'Elective:':4,tableelective[numberoftable,numberofparti
cipant]:4,'---');
            end;
```

```
            writeln;
            readln(i);
            clrscr;
            until i='next';
            final:=final+1;
        end;
    end;

Procedure Save_SeatingPlan;
begin
assign(d,'seating_plan.txt');
writeln('---');
rewrite(d);
final:=1;
for numberoftable:=1 to (finaltable) do
begin
writeln(d,'Table',numberoftable);
for numberofparticipant:=1 to (finalnumber[final]) do
writeln(d,'Seat':4,tabledata[numberoftable,numberofpar
```

```
ticipant],'Name:':4,tablename[numberoftable,numberof
participant],'Year of
graduation:':4,tableyear[numberoftable,numberofpartici
pant],'Sex:':4,tablesex[numberoftable,numberofparticip
ant],'Elective:':4,tableelective[numberoftable,numberof
participant]);
writeln(d);
final:=final+1;
end;
close(d)
end;


Procedure GenerateByElective;
var
search,again,p:integer;
full,stop,sit:boolean;
begin
clrscr;
```

```
writeln('Generating the seating plan...');
GroupingSameElective;
Table_Seat;
stop:=false;
group1sit:=0;
group2sit:=0;
group3sit:=0;
group1finish:=false;
group2finish:=false;
group3finish:=false;
require:=0;
tablenumber:=1;
numberoftable:=1;
numberofparticipant:=0;
for final :=1 to 250 do
finalnumber[final]:=0;
final:=1;
finaltable:=1;
full:=false;
```

```
if group1number=1
then begin
   group1finish:=true;
   group1sit:=1;
end;
if group2number=1
then begin
   group2finish:=true;
   group2sit:=1;
end;
if group3number=1
then begin
   group3finish:=true;
   group3sit:=1;
end;
f:=1;
g:=1;
h:=1;
p:=1;
```

```
while (group1finish) and (group2finish) and
(group3finish)=false do
begin
f:=1;
g:=1;
h:=1;
p:=1;
stop:=false;
require:=0;
full:=false;
while(fgroup[f]=true) and (f<group1number-1) do
f:=f+1;
while(ggroup[g]=true) and (g<group2number-1) do
g:=g+1;
while(hgroup[h]=true) and (h<group3number-1) do
h:=h+1;
sit:=false;
while tablefull[tablenumber]=false do
begin
```

```
    stop:=false;
    if(f<=group1number) and (fgroup[f]=false) and
(group1finish=false)
    then search:=group1data[f]
    else if(g<=group2number) and (ggroup[g]=false) and
(group2finish=false)
    then search:=groupdata[g]
    else if(h<=group3number) and (hgroup[h]=false) and
(group3finish=false)
    then search:=group3data[h];
quarryfound:=false;
writeln(search);
readln;
if (group1sit<group1number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
                then begin
```

```
                    if((readseat[search]=space) and
(loaddatause[search]=false))
                    then begin

                        quarryfound:=true;
                        tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];
```

```
tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                fgroup[f]:=true;
                                group1sit:=group1sit+1;
                                loaddatause[search]:=true;
                                if (group1sit+1)=group1number
                                then group1sit:=group1sit+1;
                                end;
                            if (readseat[search]<space) and
(loaddatause[search]=false)
                                then begin
```

```
                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
require:=space-readseat[search];
                                    fgroup[f]:=true;
                                    group1sit:=group1sit+1;
                                    loaddatause[search]:=true;
                                    end;
                          end
                          else begin
                                    if((readseat[search]=space) and
(loaddatause[search]=false))
                                    then begin
                                              quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
```

```
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
```

```
loaddatause[search]:=true;
                                    fgroup[f]:=true;
                                    group1sit:=group1sit+1;
                                    if
(group1sit+1)=group1number
                                    then
group1sit:=group1sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];
```

```
tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
                                        fgroup[f]:=true;

loaddatause[search]:=true;

group1sit:=group1sit+1;
                                        end;
```

```
                                        end;
                        end
                        else if (require=readseat[search]) and
(loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];
```

```
tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    fgroup[f]:=true;
                                    group1sit:=group1sit+1;

loaddatause[search]:=true;

if(group1sit+1)=group1number
```

```
                                    then
group1sit:=group1sit+1;
                                    end
                            else if (require>readseat[search])
and (loaddatause[search]=false)
                                    then begin
                                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
```

```
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=require-readseat[search];
                                    fgroup[f]:=true;
                                    group1sit:=group1sit+1;

loaddatause[search]:=true;
                                    end;
                                if f=(group1number-1)
                                then full:=true;
                                if (quarryfound=false) and
(group1sit+1=group1number) and (full)
                                then begin

tablefull[tablenumber]:=true;
```

```
finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    group1sit:=group1sit+1;
                                    end
                            else if (quarryfound=false) and (full)
                                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    if
```

```
(group1sit+1)=group1number
                                    then
group1sit:=group1sit+1;
                                    end;
                                if f<>(group1number-1)
                                then f:=f+1;
                                stop:=true;
                            end;
if (group2sit<group2number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin
                                quarryfound:=true;
                                tablefull[tablenumber]:=true;
```

```
numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


finalnumber[final]:=numberofparticipant;
```

```
numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                ggroup[g]:=true;
                                group2sit:=group2sit+1;
                                loaddatause[search]:=true;
                                if (group2sit+1)=group2number
                                then group2sit:=group2sit+1;
                                end;
                            if (readseat[search]<space) and
(loaddatause[search]=false)
                            then begin
                                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
                                ggroup[g]:=true;
                                group2sit:=group2sit+1;
                                loaddatause[search]:=true;
                                end;
```

```
                    end
                    else begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin
                            quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];
```

```
tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;

loaddatause[search]:=true;
                                    ggroup[g]:=true;
                                    group2sit:=group2sit+1;
                                    if
(group2sit+1)=group2number
```

```
                                    then
group2sit:=group2sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
                                then begin
                                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
```

```
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
                                        ggroup[g]:=true;

loaddatause[search]:=true;

group2sit:=group2sit+1;
                                        end;
                                        end;
                    end
                    else if (require=readseat[search]) and
(loaddatause[search]=false)
                            then begin
```

```
                                    quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    ggroup[g]:=true;
                                    group2sit:=group2sit+1;

loaddatause[search]:=true;

if(group2sit+1)=group2number
                                    then
group2sit:=group2sit+1;
                                    end
                          else if (require>readseat[search])
and (loaddatause[search]=false)
```

```
                then begin
                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
require:=require-readseat[search];
                                    ggroup[g]:=true;
                                    group2sit:=group2sit+1;

loaddatause[search]:=true;
                                    end;
                                if g=(group2number-1)
                                then full:=true;
                                if (quarryfound=false) and
(group2sit+1=group2number) and (full)
                                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
```

```
                    final:=final+1;
                    numberofparticipant:=0;
                    group2sit:=group2sit+1;
                    end
            else if (quarryfound=false) and (full)
                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                    final:=final+1;
                    numberofparticipant:=0;
                    if
(group2sit+1)=group2number
                    then
group2sit:=group2sit+1;
                    end;
```

```
                    if g<>(group2number-1)
                    then g:=g+1;
                    stop:=true;
                end;

if (group3sit<group3number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then
                        begin
                                quarryfound:=true;
                                tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;
```

```
tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
```

```
numberoftable:=numberoftable+1;
                    final:=final+1;
                    numberofparticipant:=0;
                    hgroup[h]:=true;
                    group3sit:=group3sit+1;
                    loaddatause[search]:=true;
                    if (group3sit+1)=group3number
                    then group3sit:=group3sit+1;
                    end;
                if (readseat[search]<space) and
(loaddatause[search]=false)
                    then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
```

```
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
                                hgroup[h]:=true;
                                group3sit:=group3sit+1;
                                loaddatause[search]:=true;
                                end;
                        end
```

```
                    else begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin
                            quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
```

```
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;

loaddatause[search]:=true;
                                    hgroup[h]:=true;
                                    group3sit:=group3sit+1;
                                    if
(group3sit+1)=group3number
                                    then
```

```
group3sit:=group3sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
                                then begin
                                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];
```

```
tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=space-readseat[search];
                                        hgroup[h]:=true;

loaddatause[search]:=true;

group3sit:=group3sit+1;
                                        end;
                                        end;
                    end
                    else if (require=readseat[search]) and
(loaddatause[search]=false)
                            then begin
                                quarryfound:=true;
```

```
tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    hgroup[h]:=true;
                                    group3sit:=group3sit+1;

loaddatause[search]:=true;

if(group3sit+1)=group3number
                                    then
group3sit:=group3sit+1;
                                    end
                          else if (require>readseat[search])
and (loaddatause[search]=false)
                                  then begin
                                     quarryfound:=true;
```

```
numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];

require:=require-readseat[search];
```

```
                    hgroup[h]:=true;
                    group3sit:=group3sit+1;

loaddatause[search]:=true;
                    end;
                if h=(group3number-1)
                then full:=true;
                if (quarryfound=false) and
(group3sit+1=group3number) and (full)
                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                    final:=final+1;
                    numberofparticipant:=0;
                    group3sit:=group3sit+1;
```

```
                    end
                else if (quarryfound=false) and (full)
                    then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                    final:=final+1;
                    numberofparticipant:=0;
                    if
(group3sit+1)=group3number
                    then
group3sit:=group3sit+1;
                    end;
                if h<>(group1number-1)
                then h:=h+1;
                stop:=true;
```

```
                end;
            end;
            GroupSit_Check;
            tablenumber:=tablenumber+1;
            end;
            if (group1finish and group2finish and
group3finish = true)
            then begin
                finaltable:=numberoftable-1;
                end;
            writeln('Seating plan is generated!');
            Save_SeatingPlan;

            OutputSeatingPlan;

            end;

Procedure GenerateBySex;
var
```

```
search,p,again:integer;
full,stop,sit:boolean;
begin
clrscr;
writeln('Generating the seating plan...');
GroupingSameSex;
Table_Seat;
stop:=false;
group4sit:=0;
group5sit:=0;
group4finish:=false;
group5finish:=false;
tablenumber:=1;
numberoftable:=1;
numberofparticipant:=0;
for final :=1 to 250 do
finalnumber[final]:=0;
final:=1;
finaltable:=1;
```

```
full:=false;
if group4number=1
then begin
    group4finish:=true;
    group4sit:=1;
end;
if group5number=1
then begin
    group5finish:=true;
    group5sit:=1;
end;
k:=1;
j:=1;
p:=1;
while (group4finish) and (group5finish)=false do
begin
k:=1;
j:=1;
p:=1;
```

```
stop:=false;
require:=0;
full:=false;
while(kgroup[k]=true) and (k<group4number-1) do
k:=k+1;
while(jgroup[j]=true) and (j<group5number-1) do
j:=j+1;
sit:=false;
while tablefull[tablenumber]=false do
begin
   stop:=false;
   if(k<=group4number) and (kgroup[k]=false) and
(group4finish=false)
   then search:=group4data[k]
   else if(j<=group5number) and (jgroup[j]=false) and
(group5finish=false)
   then search:=groupdata[j];
quarryfound:=false;
writeln(search);
```

```
readln;
if (group4sit<group4number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin

                                quarryfound:=true;
                                tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
```

```
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                kgroup[k]:=true;
                                group4sit:=group4sit+1;
```

```
                    loaddatause[search]:=true;
                    if (group4sit+1)=group4number
                    then group4sit:=group4sit+1;
                    end;
                if (readseat[search]<space) and
(loaddatause[search]=false)
                    then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];
```

```
tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=space-readseat[search];
                                kgroup[k]:=true;
                                group4sit:=group4sit+1;
                                loaddatause[search]:=true;
                                end;
                        end
                        else begin
                                if((readseat[search]=space) and
(loaddatause[search]=false))
                                then begin
                                    quarryfound:=true;
```

```
tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;

loaddatause[search]:=true;
                                    kgroup[k]:=true;
                                    group4sit:=group4sit+1;
                                    if
(group4sit+1)=group4number
                                    then
group4sit:=group4sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;
```

```
numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
require:=space-readseat[search];
                                        kgroup[k]:=true;

loaddatause[search]:=true;

group4sit:=group4sit+1;
                                        end;
                                        end;
                        end
                        else if (require=readseat[search]) and
(loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
```

```
                                    final:=final+1;
                                    kgroup[k]:=true;
                                    group4sit:=group4sit+1;

loaddatause[search]:=true;

if(group4sit+1)=group4number
                                    then
group4sit:=group4sit+1;
                                    end
                          else if (require>readseat[search])
and (loaddatause[search]=false)
                                then begin
                                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=require-readseat[search];
                                    kgroup[k]:=true;
                                    group4sit:=group4sit+1;

loaddatause[search]:=true;
                                    end;
```

```
                    if k=(group4number-1)
                    then full:=true;
                    if (quarryfound=false) and
(group1sit+1=group1number) and (full)
                    then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                        final:=final+1;
                        numberofparticipant:=0;
                        group4sit:=group4sit+1;
                        end
                else if (quarryfound=false) and (full)
                    then begin

tablefull[tablenumber]:=true;
```

```
finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    if
(group4sit+1)=group4number
                                    then
group4sit:=group4sit+1;
                                    end;
                                if f<>(group4number-1)
                                then k:=k+1;
                                stop:=true;
                        end;
if (group5sit<group5number) and (stop=false)
then begin
        if require=0
        then begin
```

```
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin
                                quarryfound:=true;
                                tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
```

```
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                jgroup[j]:=true;
                                group5sit:=group5sit+1;
                                loaddatause[search]:=true;
                                if (group5sit+1)=group5number
                                then group5sit:=group5sit+1;
                                end;
                            if (readseat[search]<space) and
(loaddatause[search]=false)
```

```
                then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
require:=space-readseat[search];
                                    jgroup[j]:=true;
                                    group5sit:=group5sit+1;
                                    loaddatause[search]:=true;
                                    end;
                        end
                        else begin
                                if((readseat[search]=space) and
(loaddatause[search]=false))
                                then begin
                                    quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
```

```
                                numberofparticipant:=0;

loaddatause[search]:=true;
                                jgroup[j]:=true;
                                group5sit:=group5sit+1;
                                if
(group5sit+1)=group5number
                                then
group5sit:=group5sit+1;
                                end;
                            if (readseat[search]<space)
and (loaddatause[search]=false)
                            then begin
                                quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
                                        jgroup[j]:=true;

loaddatause[search]:=true;
```

```
group5sit:=group5sit+1;
                                        end;
                                        end;
                        end
                        else if (require=readseat[search]) and
(loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
```

```
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    jgroup[j]:=true;
                                    group5sit:=group5sit+1;

loaddatause[search]:=true;
```

```
if(group5sit+1)=group5number
                                    then
group5sit:=group5sit+1;
                                    end
                          else if (require>readseat[search])
and (loaddatause[search]=false)
                                  then begin
                                     quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];
```

```
tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=require-readseat[search];
                                    jgroup[j]:=true;
                                    group5sit:=group5sit+1;

loaddatause[search]:=true;
                                    end;
                                if j=(group5number-1)
                                then full:=true;
                                if (quarryfound=false) and
(group5sit+1=group2number) and (full)
```

```
                    then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                group5sit:=group5sit+1;
                                end
                        else if (quarryfound=false) and (full)
                            then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
```

```
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    if
(group5sit+1)=group5number
                                    then
group5sit:=group5sit+1;
                                    end;
                                if j<>(group5number-1)
                                then j:=j+1;
                                stop:=true;
                            end;
                        end;
                        GroupSit_Check2;
                        tablenumber:=tablenumber+1;
                        end;
                        if (group4finish and group5finish =
true)
                        then begin
                                finaltable:=numberoftable-1;
```

```
            end;
        writeln('Seating plan is generated!');
        Save_SeatingPlan;

        OutputSeatingPlan;

        end;

Procedure GenerateByYear;
var
search,p,again:integer;
full,stop,sit:boolean;
begin
clrscr;
writeln('Generating the seating plan...');
GroupingSameYearOfGraduation;
Table_Seat;
stop:=false;
group6sit:=0;
```

```
group7sit:=0;
group8sit:=0;
group9sit:=0;
group6finish:=false;
group7finish:=false;
group8finish:=false;
group9finish:=false;
require:=0;
tablenumber:=1;
numberoftable:=1;
numberofparticipant:=0;
for final :=1 to 250 do
finalnumber[final]:=0;
final:=1;
finaltable:=1;
full:=false;
if group6number=1
then begin
    group6finish:=true;
```

```
    group6sit:=1;
end;
if group7number=1
then begin
    group7finish:=true;
    group7sit:=1;
end;
if group8number=1
then begin
    group8finish:=true;
    group8sit:=1;
end;
if group9number=1
then begin
    group9finish:=true;
    group9sit:=1;
end;
l:=1;
m:=1;
```

```
n:=1;
o:=1;
p:=1;
while (group6finish) and (group7finish) and
(group8finish) and (group9finish)=false do
begin
l:=1;
m:=1;
n:=1;
o:=1;
p:=1;
stop:=false;
require:=0;
full:=false;
while(lgroup[l]=true) and (l<group6number-1) do
l:=l+1;
while(mgroup[m]=true) and (m<group7number-1) do
m:=m+1;
while(ngroup[n]=true) and (n<group8number-1) do
```

```
n:=n+1;
while(ogroup[o]=true) and (o<group9number-1) do
o:=o+1;
sit:=false;
while tablefull[tablenumber]=false do
begin
   stop:=false;
   if(l<=group6number) and (lgroup[l]=false) and
(group6finish=false)
   then search:=group6data[l]
   else if(m<=group7number) and (mgroup[m]=false)
and (group7finish=false)
   then search:=group7data[m]
   else if(n<=group8number) and (ngroup[n]=false) and
(group8finish=false)
   then search:=group8data[n]
   else if(o<=group9number) and (ogroup[o]=false) and
(group9finish=false)
   then search:=group9data[o];
```

```
quarryfound:=false;
writeln(search);
readln;
if (group6sit<group6number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin

                                quarryfound:=true;
                                tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                        final:=final+1;
                        numberofparticipant:=0;
```

```
                    lgroup[l]:=true;
                    group6sit:=group6sit+1;
                    loaddatause[search]:=true;
                    if (group6sit+1)=group6number
                    then group6sit:=group6sit+1;
                    end;
                if (readseat[search]<space) and
(loaddatause[search]=false)
                    then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
```

```
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=space-readseat[search];
                                lgroup[l]:=true;
                                group6sit:=group6sit+1;
                                loaddatause[search]:=true;
                                end;
                        end
                        else begin
                                if((readseat[search]=space) and
(loaddatause[search]=false))
                                then begin
```

```
                                    quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;

loaddatause[search]:=true;
                                    lgroup[l]:=true;
                                    group6sit:=group6sit+1;
                                    if
(group6sit+1)=group6number
                                    then
group6sit:=group6sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
```

```
                then begin
                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
require:=space-readseat[search];
                                        lgroup[l]:=true;

loaddatause[search]:=true;

group6sit:=group6sit+1;
                                        end;
                                        end;
                        end
                        else if (require=readseat[search]) and
(loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;
```

```
tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                    numberofparticipant:=0;
```

```
numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    lgroup[l]:=true;
                                    group6sit:=group6sit+1;

loaddatause[search]:=true;

if(group6sit+1)=group6number
                                    then
group6sit:=group6sit+1;
                                    end
                          else if (require>readseat[search])
and (loaddatause[search]=false)
                                then begin
                                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;
```

```
tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=require-readseat[search];
                                    lgroup[l]:=true;
                                    group6sit:=group6sit+1;
```

```
loaddatause[search]:=true;
                                    end;
                                if l=(group6number-1)
                                then full:=true;
                                if (quarryfound=false) and
(group6sit+1=group6number) and (full)
                                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    group6sit:=group6sit+1;
                                    end
                            else if (quarryfound=false) and (full)
                                then begin
```

```
tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    if
(group6sit+1)=group6number
                                    then
group6sit:=group6sit+1;
                                    end;
                               if l<>(group6number-1)
                               then l:=l+1;
                               stop:=true;
                          end;
if (group7sit<group7number) and (stop=false)
then begin
```

```
        if require=0
        then begin
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then begin
                                quarryfound:=true;
                                tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];
```

```
tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                        final:=final+1;
                        numberofparticipant:=0;
                        mgroup[m]:=true;
                        group7sit:=group7sit+1;
                        loaddatause[search]:=true;
                        if (group7sit+1)=group7number
                        then group7sit:=group7sit+1;
                        end;
```

```
                    if (readseat[search]<space) and
(loaddatause[search]=false)
                    then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
```

```
delective[search];



require:=space-readseat[search];
                                    mgroup[m]:=true;
                                    group7sit:=group7sit+1;
                                    loaddatause[search]:=true;
                                    end;
                        end
                        else begin
                                if((readseat[search]=space) and
(loaddatause[search]=false))
                                then begin
                                    quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;
```

```
tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
```

```
numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;

loaddatause[search]:=true;
                                    mgroup[m]:=true;
                                    group7sit:=group7sit+1;
                                    if
(group7sit+1)=group7number
                                    then
group7sit:=group7sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;
```

```
tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
                                    mgroup[m]:=true;
```

```
loaddatause[search]:=true;

group7sit:=group7sit+1;
                                        end;
                                        end;
                        end
                        else if (require=readseat[search]) and
(loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];
```

```
tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    mgroup[m]:=true;
                                    group7sit:=group7sit+1;
```

```
loaddatause[search]:=true;

if(group7sit+1)=group7number
                                    then
group7sit:=group7sit+1;
                                    end
                          else if (require>readseat[search])
and (loaddatause[search]=false)
                                  then begin
                                     quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];
```

```
tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=require-readseat[search];
                                    mgroup[m]:=true;
                                    group7sit:=group7sit+1;

loaddatause[search]:=true;
                                    end;
                                if m=(group7number-1)
                                then full:=true;
```

```
                    if (quarryfound=false) and
(group7sit+1=group7number) and (full)
                    then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                        final:=final+1;
                        numberofparticipant:=0;
                        group7sit:=group7sit+1;
                        end
                else if (quarryfound=false) and (full)
                    then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;
```

```
numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    if
(group7sit+1)=group7number
                                    then
group7sit:=group7sit+1;
                                    end;
                                if m<>(group7number-1)
                                then m:=m+1;
                                stop:=true;
                        end;

if (group8sit<group8number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
```

```
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then
                        begin
                                quarryfound:=true;
                                tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
```

```
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                ngroup[n]:=true;
                                group8sit:=group8sit+1;
                                loaddatause[search]:=true;
                                if (group8sit+1)=group8number
                                then group8sit:=group8sit+1;
                                end;
                            if (readseat[search]<space) and
```

```
(loaddatause[search]=false)
                    then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
require:=space-readseat[search];
                                    ngroup[n]:=true;
                                    group8sit:=group8sit+1;
                                    loaddatause[search]:=true;
                                    end;
                          end
                          else begin
                                    if((readseat[search]=space) and
(loaddatause[search]=false))
                                    then begin
                                              quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;
```

```
tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
```

```
                    final:=final+1;
                    numberofparticipant:=0;

loaddatause[search]:=true;
                    ngroup[n]:=true;
                    group8sit:=group8sit+1;
                    if
(group8sit+1)=group8number
                    then
group8sit:=group8sit+1;
                    end;
                if (readseat[search]<space)
and (loaddatause[search]=false)
                then begin
                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;
```

```
tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=space-readseat[search];
                                        ngroup[n]:=true;

loaddatause[search]:=true;
```

```
group8sit:=group8sit+1;
                                        end;
                                        end;
                        end
                        else if (require=readseat[search]) and
(loaddatause[search]=false)
                                then begin
                                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
```

```
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    ngroup[n]:=true;
                                    group8sit:=group8sit+1;

loaddatause[search]:=true;
```

```
if(group8sit+1)=group8number
                                    then
group8sit:=group8sit+1;
                                    end
                            else if (require>readseat[search])
and (loaddatause[search]=false)
                                    then begin
                                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];
```

```
tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=require-readseat[search];
                                        ngroup[n]:=true;
                                        group8sit:=group8sit+1;

loaddatause[search]:=true;
                                        end;
                                    if n=(group8number-1)
                                    then full:=true;
                                    if (quarryfound=false) and
(group8sit+1=group8number) and (full)
                                    then begin
```

```
tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    group8sit:=group8sit+1;
                                    end
                            else if (quarryfound=false) and (full)
                                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
```

```
                                        numberofparticipant:=0;
                                        if
(group8sit+1)=group8number
                                        then
group8sit:=group8sit+1;
                                        end;
                                    if n<>(group1number-1)
                                    then n:=n+1;
                                    stop:=true;
                                end;
if (group9sit<group9number) and (stop=false)
then begin
        if require=0
        then begin
                if space<10
                then begin
                        if((readseat[search]=space) and
(loaddatause[search]=false))
                        then
```

```
                    begin
                        quarryfound:=true;
                        tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];
```

```
finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                final:=final+1;
                                numberofparticipant:=0;
                                ogroup[o]:=true;
                                group9sit:=group9sit+1;
                                loaddatause[search]:=true;
                                if (group9sit+1)=group9number
                                then group9sit:=group9sit+1;
                                end;
                            if (readseat[search]<space) and
(loaddatause[search]=false)
                                then begin
                                     quarryfound:=true;
```

```
numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



require:=space-readseat[search];
```

```
                    ogroup[o]:=true;
                    group9sit:=group9sit+1;
                    loaddatause[search]:=true;
                    end;
                end
                else begin
                    if((readseat[search]=space) and
(loaddatause[search]=false))
                    then begin
                        quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];
```

```
tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;

loaddatause[search]:=true;
```

```
                                    ogroup[o]:=true;
                                    group9sit:=group9sit+1;
                                    if
(group9sit+1)=group9number
                                    then
group9sit:=group9sit+1;
                                    end;
                                if (readseat[search]<space)
and (loaddatause[search]=false)
                                then begin
                                    quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];
```

```
tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];

tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=space-readseat[search];
                                        ogroup[o]:=true;

loaddatause[search]:=true;

group9sit:=group9sit+1;
                                        end;
                                        end;
                        end
```

```
                    else if (require=readseat[search]) and
(loaddatause[search]=false)
                            then begin
                                quarryfound:=true;

tablefull[tablenumber]:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];
```

```
tableelective[numberoftable,numberofparticipant]:=rea
delective[search];



finalnumber[final]:=numberofparticipant;
                                    numberofparticipant:=0;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    ogroup[o]:=true;
                                    group9sit:=group9sit+1;

loaddatause[search]:=true;

if(group9sit+1)=group9number
                                    then
group9sit:=group9sit+1;
```

```
                    end
                else if (require>readseat[search])
and (loaddatause[search]=false)
                    then begin
                        quarryfound:=true;

numberofparticipant:=numberofparticipant+1;

tabledata[numberoftable,numberofparticipant]:=search;

tablename[numberoftable,numberofparticipant]:=readn
ame[search];

tableyear[numberoftable,numberofparticipant]:=readye
ar[search];

tablesex[numberoftable,numberofparticipant]:=readsex[
search];
```

```
tableelective[numberoftable,numberofparticipant]:=rea
delective[search];


require:=require-readseat[search];
                                    ogroup[o]:=true;
                                    group9sit:=group9sit+1;

loaddatause[search]:=true;
                                    end;
                                if o=(group9number-1)
                                then full:=true;
                                if (quarryfound=false) and
(group9sit+1=group9number) and (full)
                                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;
```

```
numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    group9sit:=group9sit+1;
                                    end
                            else if (quarryfound=false) and (full)
                                then begin

tablefull[tablenumber]:=true;

finalnumber[final]:=numberofparticipant;

numberoftable:=numberoftable+1;
                                    final:=final+1;
                                    numberofparticipant:=0;
                                    if
(group9sit+1)=group9number
                                    then
```

```
group9sit:=group9sit+1;
                                        end;
                                if o<>(group9number-1)
                                then o:=o+1;
                                stop:=true;
                            end;
                        end;
                        GroupSit_Check3;
                        tablenumber:=tablenumber+1;
                        end;
                        if (group6finish and group7finish and
group8finish and group9finish = true)
                        then begin
                                finaltable:=numberoftable-1;
                                end;
                        writeln('Seating plan is generated!');
                        Save_SeatingPlan;

                        OutputSeatingPlan;
```

```
                    end;

Procedure GenerateSeatingPlan;
begin
   clrscr;
   writeln('Please select the segement og the seating
plan.');
   writeln('Press 1 for generating seating plan by
elective.');
   writeln('Press 2 for generating seating plan by sex.');
   writeln('Press 3 for generating seating plan by year of
graduation.');
   readln(k);
   case k of
   1:GenerateByElective;
   2:GenerateBySex;
   3:GenerateByYear;
   end;
```

```
end;


begin
   clrscr;
   textcolor(yellow);
   textbackground(magenta);
   assign(data,'data.txt');
   reset(data);
   close(data);
   repeat
      writeln('------------------------------');
      writeln('Press 1 for entering information.');
      writeln('Press 2 for changing information.');
      writeln('Press 3 for deleting information.');
      writeln('Press 4 for generating seat plan.');
      writeln('Press 5 for leaving program.');
      writeln('------------------------------');
      readln(choice);
      case choice of
```

```
        1:EnterInformation;
        2:ChangeInformation;
        3:DeleteInformation;
        4:GenerateSeatingPlan;
        5:begin
            writeln('Press any key to leave');
            readln;
          end;
            else
                writeln('Syntax error!');
            end;
          until choice=5;
end.
```

Working Schedule

July,2017	Chapter.1 Background + Define objectives + Propose Functions
August,2017	Chapter.2 Design
September,2017	Chapter.3 Implementation
October,2017	Chapter 5: Reference & Acknowledgement

November,2017	Chapter.4 Testing & Evaluation + Finish the final program
December,2017	Finish the final project