Form 6 ICT SBA – Case Study 2-1

Multiple-Choice Marking

Form 6 ______ Name ______________________________ No. ______ Date___________

A program `MCMark.pas` is written to check the correctness of a multiple-choice test. A file `AnsKey.txt` stores the answer keys of the test while another file, e.g. `Peter.txt`, stores the answers of a student (Peter).

In these files,

(a) the answer of each question is stored in separate lines;

(b) blank lines are nor allowed;

(c) number of answers in student's file is equal to that of the file `AnsKey.txt`.

Examples of these files are shown below.

```
Contents of AnsKey.txt:
A
C
E
B
B
A
D
:
```

```
Contents of Peter.txt:
A
D
C
B
B
A
E
:
```

Sample output

```
Enter the file name: Peter.txt
Number of questions: 12
Number of correct answers: 7
Percentage: 58.33%
```

Complete the following Pascal program `MCMark.pas`.

```
program MCMarking;
var
  K, S : text;
  filename : string;
  KAns, SAns : char;      { KAns = Answer Key, SAns = Student's Answer }
  numQ, numCor : integer;  {numQ = number of questions, numCor = number of correct ans.}
begin
  write('Enter the file name: ');
  readln(filename);
  assign(K, 'AnsKey.txt');
  assign(____________________);
  reset(K);
  ____________________;
  numQ := 0;
  numCor := 0;
  while not eof(K) do
    begin
      readln(K, _______);
      readln(S, SAns);
      numQ := _________________;
      if KAns = _________ then
          numCor := _______________________
    end;
  close(K);
  close(S);
  writeln('Number of questions: ', numQ);
  writeln('Number of correct answers: ', numCor);
  writeln('Correct Percent: ', _________________________________);
end.
```

Form 6 ICT SBA – Case Study 2-2

True/False Marking

Form 6 ______ Name _______________________________ No. ______ Date___________

Background

An ICT teacher has designed 10 sets of True/False (T/F) exercises for his students to practice. In each T/F exercise, there are 50 questions. The answer of each question is ‘T’ or ‘F’ only.

There are 100 students taking the ICT subject. Each of them has to complete the 10 sets of exercises. The ICT teacher would like to use a Pascal program to mark all the T/F answers of his students.

The answer keys for the 10 sets of exercises are stored in 10 text files: ANS01.TXT, ANS02.TXT, …, ANS10.TXT; whereas the answers of his students for the 10 exercises are stored in 10 text files: TFDATA01.TXT, TFDATA02.TXT, …, TFDATA10.TXT respectively.

The following examples show the formats of the text files.

For the answer keys of exercise ## (ANS##.TXT):

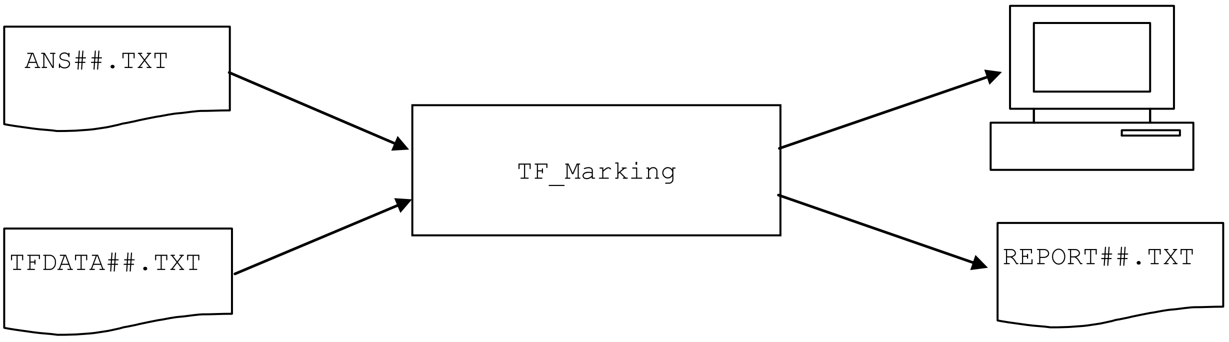
```
TFTTFFTFTTFFFFTFTFTFTFTTFFTFTTFFFFTFTFTFTFTFTFTFTT
```

For the answers of the students in the exercise ## (TFDATA##.TXT):

```
4A12FFTTFTTFTTFFFFFFTFTTTFTTFFTFTTFFFFTFTFTFTFTFTFTFFF
……………………………..
4C34FFFFFFTFTTTTTFFFFTFTTFTFTFTFTFTTTFFTFTTFFFFTFTFFTTT
```

class class no. 50 answers

Data Flow:

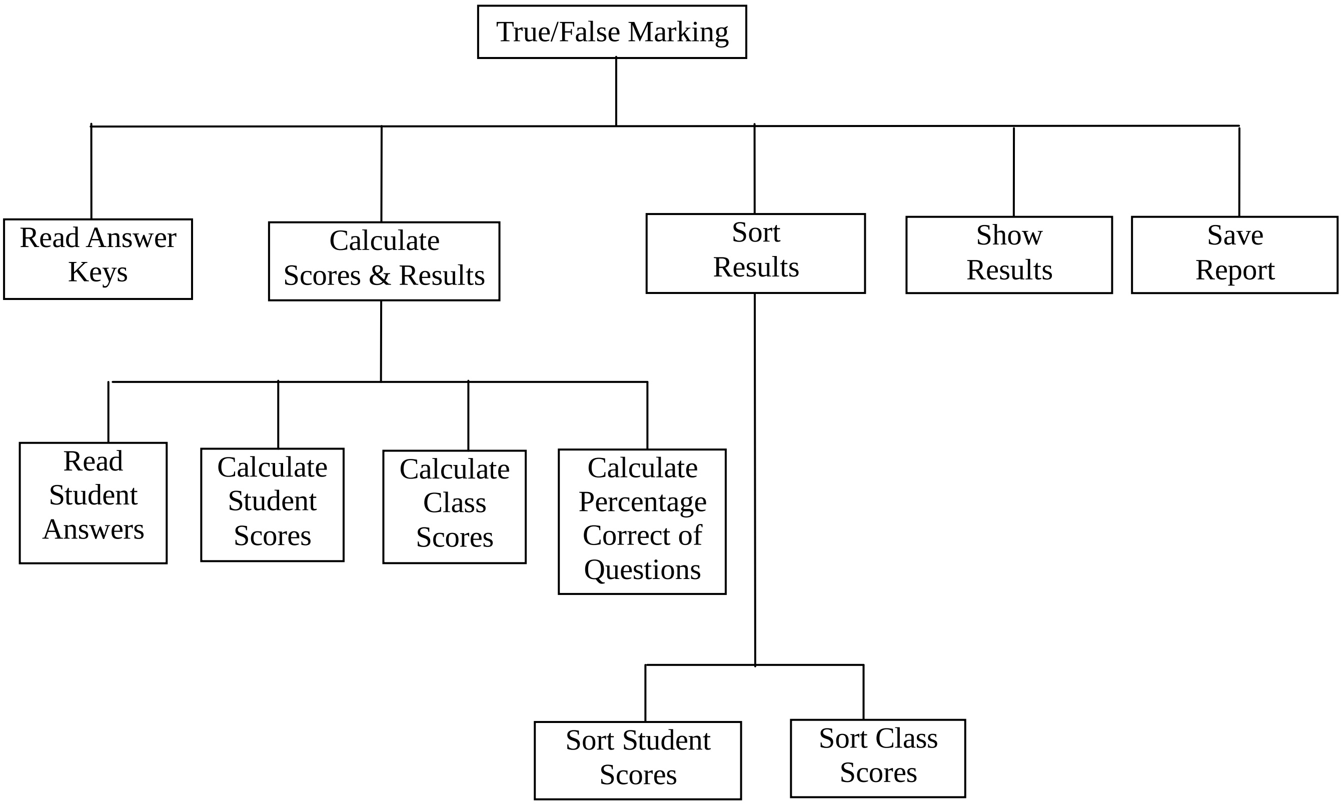


The report file (`REPORT##.TXT`) for each exercise consists of the score (out of 50) of each student, the average score of each class, and the percentage correct of each question, etc. All the scores should be sorted in descending order.

Suppose that you were the ICT teacher. Design a Pascal program to complete the above tasks.

Design of Solution

The problem can be divided into some sub-problems as shown in the figure below:



The layout of the report is shown below:

```
<< Report on T/F Questions >>

*****************************************************
Question Analysis

Percentage correct of each question:
 Q1 16.13%   Q2 24.73%   Q3 22.58%   Q4 22.58%   Q5 29.03%
........................................................
........................................................
........................................................
Q46 21.51%  Q47 24.73%  Q48 19.35%  Q49 16.13%  Q50 21.51%

******************************************************
Class Results:

Total number of classes: 5

Average score of classes in descending order:
Class  Size  Average Score
   4B    23        25.22
   4E    18        24.83
  ..............
  ..............

*****************************************************
Student Results:

Total number of students: 100

Scores of students in descending order:
Class  No.  Score
   4B   26    33
   4B   12    31
  ..............
  ..............
```

Data Structures

The following data structures will be adopted in the program:

- For storing the answers read from the "answer keys" file
  - A 1-dimensional array:
    ```
    answer : string[questno];
    ```
- For storing the class, class number read from the "student answers" file and the scores of the corresponding student:
  - Parallel arrays:
    ```
    stud_class : array[1..maxstudno] of string[2];
    stud_no : array[1..maxstudno] of string[2];
    stud_score : array[1..maxstudno] of integer;
    ```
- For storing the class code, number of students and average score of each class:
  - Parallel arrays:
    ```
    class_code : array[1..maxclasssize] of string[2];
    class_size : array[1..maxclasssize] of integer;
    class_score : array[1..maxclasssize] of real;
    ```
- For storing the number of students getting the correct answer in each question:
  - A 1-dimensional array:
    ```
    quest_corr : array[1..questno] of integer;
    ```

Procedures

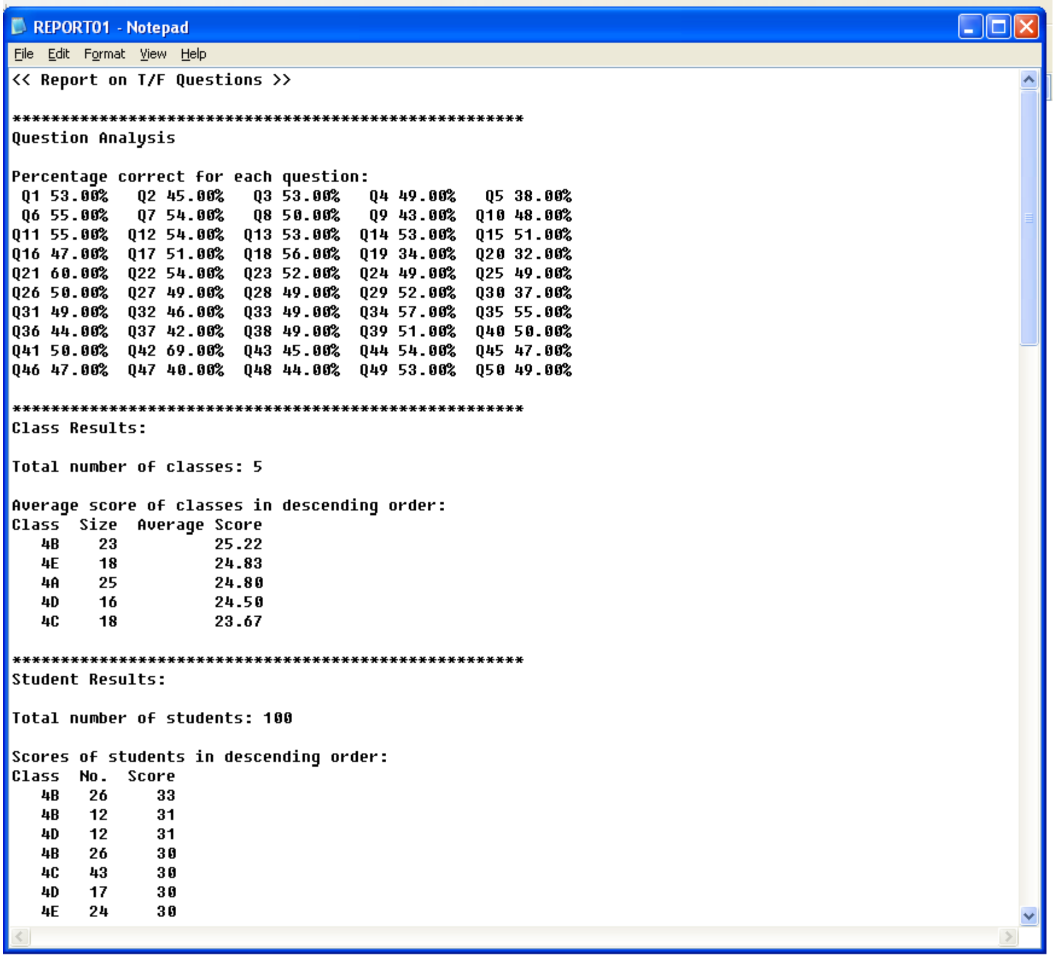
The program consists of the following main procedures

- **procedure read_ans_keys**
  - Read the answers from the file `ANS##.TXT` and store them into the array `answer[.]`.
- **procedure calculate_stud_scores**
  - Read the students' data from the file `TFDATA##.TXT` and store them into the parallel arrays `stud_class[.]` and `stud_no[.]`
  - Compare the student's answers with the answer keys
  - Store the number of correct answers of each student into the array `stud_score[.]`
  - Count the number of students getting the correct answer for each question and store the count into the array `quest_corr[.]`
- **procedure calculate_class_score**
  - Classify different class codes, count the number of students in each class and calculate the average score of each class
  - Store the corresponding results into the parallel arrays `class_code[.]`, `class_size[.]`and `class_score[.]`
- **procedure sort_stud_score**
  - Sort the student records in the parallel arrays in descending order of the student score (e.g. by using bubble sort)
- **procedure sort_class_score**
  - Sort the class records in the parallel arrays in descending order of the class average score (e.g. by using bubble sort)
- **procedure save_report**
  - Save the results into the text file `RESULT##.TXT` according to the design of the layout
- **procedure show_results**
  - Show the results on the screen according to the option selected by user: e.g. top five, bottom five, class average scores, question analysis, etc.

Sample Output

```
Enter the filename of the Answer Keys: ANS01.txt
Enter the filename of the Students' Answers: TFDATA01.txt

Enter the filename of the Report file: RERORT01.txt

File saved
Press <enter> to continue

<< Results of T/F Questions >>
*****************************

1. Show top five
2. Show bottom five
3. Show class average scores
4. Question Analysis
5. Stop program

Enter your choice (1/2/3/4/5) : 1

Top Five Results:
Class  No.  Score
   4B   26    33
   4B   12    31
   4D   12    31
   4B   26    30
   4C   43    30
*****************************************************

Press <enter> to continue
```

Report File



```
<< Report on T/F Questions >>

*****************************************************
Question Analysis

Percentage correct for each question:
 Q1 53.00%   Q2 45.00%   Q3 53.00%   Q4 49.00%   Q5 38.00%
 Q6 55.00%   Q7 54.00%   Q8 50.00%   Q9 43.00%  Q10 48.00%
Q11 55.00%  Q12 54.00%  Q13 53.00%  Q14 53.00%  Q15 51.00%
Q16 47.00%  Q17 51.00%  Q18 56.00%  Q19 34.00%  Q20 32.00%
Q21 60.00%  Q22 54.00%  Q23 52.00%  Q24 49.00%  Q25 49.00%
Q26 50.00%  Q27 49.00%  Q28 49.00%  Q29 52.00%  Q30 37.00%
Q31 49.00%  Q32 46.00%  Q33 49.00%  Q34 57.00%  Q35 55.00%
Q36 44.00%  Q37 42.00%  Q38 49.00%  Q39 51.00%  Q40 50.00%
Q41 50.00%  Q42 69.00%  Q43 45.00%  Q44 54.00%  Q45 47.00%
Q46 47.00%  Q47 40.00%  Q48 44.00%  Q49 53.00%  Q50 49.00%

*****************************************************
Class Results:

Total number of classes: 5

Average score of classes in descending order:
Class  Size  Average Score
   4B    23          25.22
   4E    18          24.83
   4A    25          24.80
   4D    16          24.50
   4C    18          23.67

*****************************************************
Student Results:

Total number of students: 100

Scores of students in descending order:
Class  No.  Score
   4B   26     33
   4B   12     31
   4D   12     31
   4B   26     30
   4C   43     30
   4D   17     30
   4E   24     30
```

** Try to run the executable file to see all the results.

Program Coding

A partly completed program is shown below. Try to complete it according to the above information.

```
program TF_Marking;
uses crt;
const  maxstudno = 200; maxclasssize = 10; questno = 50;

var
  answer : string[questno];
  stud_class : array[1..maxstudno] of string[2];
  stud_no : array[1..maxstudno] of string[2];
  stud_score : array[1..maxstudno] of integer;
  class_code : array[1..maxclasssize] of string[6];
  class_size : array[1..maxclasssize] of integer;
  class_score : array[1..maxclasssize] of real;
  quest_corr : array[1..questno] of integer;
  stud_count, class_count : integer;

procedure read_ans_keys;
var
  ans_file_name : string;
  infile : text;
begin
  write('Enter the filename of the Answer Keys: ');
  readln(ans_file_name);
  assign(infile, ans_file_name);
  reset(_____________);
  readln(_________, answer);
  close(infile)
end;

procedure calculate_stud_scores;
var
  stud_file_name : string;
  infile : text;
  stud_record : string;
  stud_ans : string[questno];
  i, j, k, corr_count : integer;
begin
  write('Enter the filename of the Students'' Answers: ');
  readln(stud_file_name);
  assign(_______________________);
  __________________;

  for k := 1 to questno do
    quest_corr[k] := 0;

  i := 0;
  while _____________________________ do
    begin
      i := i + 1;
      readln(infile, stud_record);
      stud_class[i] := copy(stud_record, 1, 2);
      stud_no[i] := copy(_____________________________);
      stud_ans   := copy(stud_record,______, questno);
      corr_count := 0;
      for j := 1 to questno do
        if stud_ans[j] = ______________ then
          begin
            corr_count := corr_count + 1;
            quest_corr[j] := ____________________
          end;
      stud_score[i] := corr_count;
    end;
  stud_count := i;

  close(infile)
end;
```

```
procedure calculate_class_score;
var
  i, j : integer;   { i = pointer for student, j = pointer for class }
  found : boolean;
begin
  class_code[1] := stud_class[1];
  class_size[1] := 1;
  class_score[1] := stud_score[1];
  class_count := 1;
  for i := 2 to stud_count do
   begin
     j := 1;
     found := false;
     repeat
       if class_code[j] = stud_class[i] then
         begin
           class_size[j] := ___________________________________________;
           class_score[j] := __________________________________________;
           found := ________________________________
         end
       else
         j := j + 1
     until found or (j > ________________);
     if not found then
       begin
         class_count := j;
         class_code[class_count] := stud_class[i];
         class_size[class_count] := 1;
         class_score[class_count] := ____________________
       end
   end;
  for i := 1 to class_count do
   class_score[i] := class_score[i]/______________________________
end;

procedure sort_stud_score;  { descending order }
var
  k, j : integer;
  tmp_class : string[2];
  tmp_no : string[2];
  tmp_score : integer;
begin
 { Bubble Sort }
  for k := 1 to stud_count - 1 do
   for j := 1 to stud_count - k do
     if stud_score[j] < stud_score[j+1] then
         begin
           tmp_class := stud_class[j];
           stud_class[j] := stud_class[j+1];
           stud_class[j+1] := tmp_class;
           tmp_no := stud_no[j];
           stud_no[j] := stud_no[j+1];
           stud_no[j+1] := tmp_no;

         end
end;
```

```
procedure sort_class_score;  { descending order }
var
  k, j : integer;
  tmp_size : integer;
  tmp_code : string[2];
  tmp_score : real;
begin
  for ______________________________________ do
    for __________________________________________ do
      if class_score[j] < class_score[j+1] then
          begin
            tmp_code := class_code[j];
            class_code[j] := class_code[j+1];
            class_code[j+1] := tmp_code;

            tmp_size := class_size[j];
            class_size[j] := class_size[j+1];
            class_size[j+1] := tmp_size;

           tmp_score := class_score[j];
            class_score[j] := class_score[j+1];
            class_score[j+1] := tmp_score;
          end
end;


procedure save_report;
var
  report_file_name : string;
  i : integer;
  outfile : text;
begin
  writeln;
  write('Enter the filename of the Report file: ');
  readln(report_file_name);
  assign(____________________________);
  _____________(outfile);
  writeln(____________, '<< Report on T/F Questions >>');
  writeln(outfile);
  writeln(outfile, '****************************************************');
  writeln(outfile, 'Question Analysis');
  writeln(outfile);
  writeln(outfile, 'Percentage correct for each question:');
  for i := 1 to 9 do
    begin
      write(outfile, ' Q', i, quest_corr[i]/_____________________________, '%  ');
      if i mod 5 = 0 then
         writeln(outfile)
    end;
  for i := 10 to questno do
    begin
      write(outfile, 'Q', i, quest_corr[i]/stud_count*100:6:2, '%  ');
      if i mod 5 = 0 then
         _________________________________
    end;
  writeln(outfile);
  writeln(outfile, '****************************************************');
  writeln(outfile, 'Class Results:');
  writeln(outfile);
  writeln(outfile, 'Total number of classes: ', ____________________);
  writeln(outfile);
  writeln(outfile, 'Average score of classes in descending order:');
  writeln(outfile, 'Class':5, 'Size':6, 'Average Score':15);
  for i := 1 to class_count do
    writeln(outfile, class_code[i]:5, __________________________, class_score[i]:15:2);
  writeln(outfile);
  writeln(outfile, '****************************************************');
  writeln(outfile, 'Student Results:');
  writeln(outfile);
  writeln(outfile, 'Total number of students: ', stud_count);
  writeln(outfile);
  writeln(outfile, 'Scores of students in descending order:');
  writeln(outfile, 'Class':5, 'No.':5, 'Score':7);
  for i := 1 to stud_count do
    writeln(outfile, stud_class[i]:5, stud_no[i]:5, ______________________);
  writeln(outfile);
  writeln(outfile, '****************************************************');
  __________________________;
  writeln;
  writeln('File saved');
  write('Press <enter> to continue');
  readln
end;

procedure show_results;
var
  choice : char;

  procedure show_top_five;
  var
    i : integer;
  begin
    writeln('Top Five Results:');
    writeln('Class':5, 'No.':5, 'Score':7);
    for i := 1 to 5 do
      writeln(stud_class[i]:5, stud_no[i]:5, stud_score[i]:7);
    writeln('****************************************************')
  end;

  procedure show_bottom_five;
  var
    i : integer;
  begin
    writeln('Bottom Five Results:');
    writeln('Class':5, 'No.':5, 'Score':7);
    for _______________________________________________________ do
      writeln(stud_class[i]:5, stud_no[i]:5, stud_score[i]:7);
    writeln('****************************************************')
  end;

  procedure show_class_average;
```

```
  procedure question_analysis;
```

```
begin  { Main program }
  repeat
    clrscr;
    writeln('<< Results of T/F Questions >>');
    writeln('*****************************');
    writeln;
    writeln('1. Show top five');
    writeln('2. Show bottom five');
    writeln('3. Show class average scores');
    writeln('4. Question Analysis ');
    writeln('5. Stop program' );
    writeln;
    write('Enter your choice (1/2/3/4/5) : ' );
    readln(choice );
    writeln;
    case choice of
      '1' : show_top_five;
      '2' : _______________________;
      '3' : _______________________;
      '4' : _______________________
    end;
    writeln;
    write('Press <enter> to continue');
    readln
  until choice = '5';
  writeln( 'Program stopped.' )
end;

begin
  ClrScr;
  read_ans_keys;

end.
```