THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

2017 SCHOLARSHIP EXAMINATION

PRACTICAL SECTION

DEPARTMENT	Computer Science
COURSE TITLE	Year 13 Scholarship
TIME ALLOWED	Six hours with a break for lunch at the discretion of the supervisor
NUMBER OF QUESTIONS IN PAPER	Three
NUMBER OF QUESTIONS TO BE ANSWERED	Three
GENERAL INSTRUCTIONS	Candidates are to answer ALL THREE questions. All questions are important. Answer as much of each question as you can. Plan your time to allow a good attempt at each question.
SPECIAL INSTRUCTIONS	Please hand in listings, notes and answers to written questions, and a Pen Drive or DVD with your program/computer work for each question. In addition please make sure that a copy of each program is printed, or stored as a plain text file. You cannot assume that the examiner has available any special software that might be required to read your files. Candidates may use any text or manual for reference during the examination. Candidates may not have access to the internet during the examination.
CALCULATORS PERMITTED	Yes

TURN OVER

1. **Polling** (Spreadsheet Use)

In this question you are asked to use a spreadsheet to do calculations and to display the results. We expect that the spreadsheet will be used for all calculations unless the question states otherwise - you will be marked down for performing calculations by hand and directly entering the results. Your work will be graded on three criteria.

(i) The accuracy of your results.

(ii) The skill you show in making use of the capabilities of the spreadsheet.

(iii) The presentation of your results. We have deliberately not provided any instructions concerning layout or formatting and our example graphs lack labels and proper scales.

In this question you are asked to use a spreadsheet to explore and graph some data. The goal is to set up your spreadsheet in such a way as to allow someone else (your customer is someone not very familiar with spreadsheets) to manipulate and draw conclusions from their data. Presenting the information in such a way as to allow this person to work easily is your task. Your spreadsheet may be used again with new data in the future.

The sample data you are to work with is a set of poll results. Imagine that your country is near the end of a general election campaign. A polling company has been phoning selected voters to ask questions every few days. In each poll 500 voters have been asked the following questions.

1. What is your gender?
   a. Male
   b. Female
2. How old are you? Select the range into which you fall.
   a. 18-24
   b. 25-34
   c. 35-49
   d. 50-64
   e. 65 or over
3. Which party do you support? There are five parties in the election. Their names are:
   a. Country Last
   b. International
   c. Belaboured
   d. Equity
   e. Yellow
4. Which of the following issues do you think is the most important?
   a. The economy
   b. Immigration
   c. The environment

Participants agreed to answer all questions, so there are no “don’t know” entries or missing entries in the poll results.

The poll data has been encoded into a csv (comma separated value) file called “Poll.csv” which is available for you as one of the exam materials supplied to your supervisor. Please note that this is not real data – it has be artificially generated for the exam.

The first five lines of the file are as follows:

```
Gender,Age,Party,Issue
a,d,c,a
b,d,c,a
b,c,a,c
```

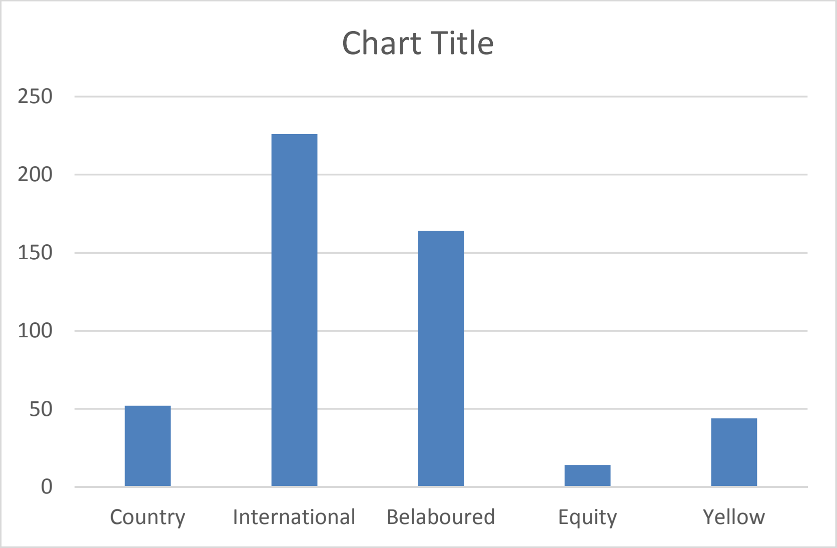
The first data line tells us that a person selected option “a” for gender (male)11 lines, was between 50 and 64 years of age, supported the “Belaboured” party, and felt that the economy was the most important issue.

Stage A

Create a spreadsheet. Load the poll results from the CSV file.

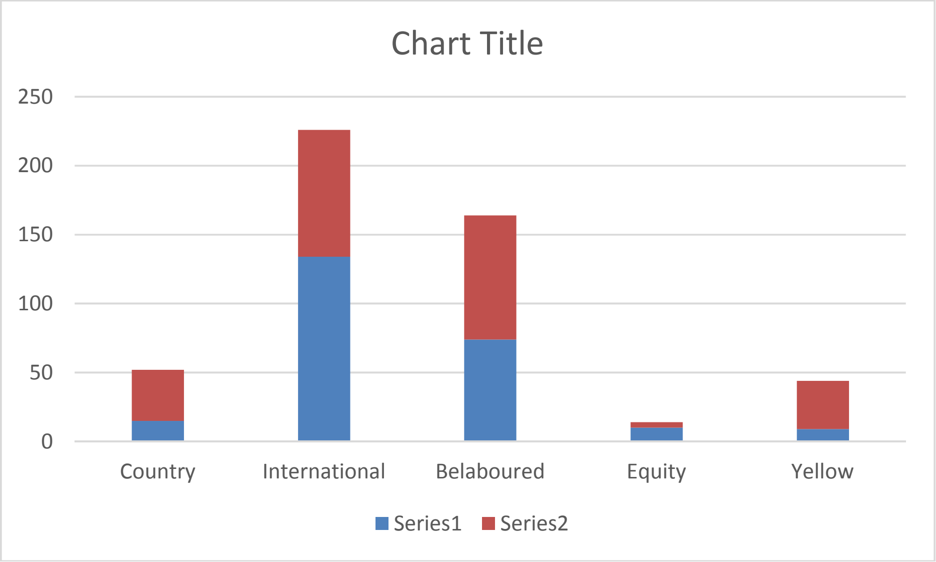
Stage B

Your first task is to produce bar charts showing the responses to each question. For example, the chart for question 3 might look a little like this: (except that it would be accurately labelled and would have a more useful y axis scale)



Stage C

Your next task is to show how the answers to questions 3 and 4 relate to the answers to questions 1 and 2. For example the chart below shows how the supporters of each party are made up by gender. Again, some work on labels and scales would be helpful.



Stage D

This task asks you to produce charts from subsets of the data. The polling company wants to be able to ask questions like: “What is the support for each of the parties from females in the age range 18 to 24”. Your task is to make that possible for them, not by coding the specific question given, but by setting up a system that will allow them to easily choose the subset they want. You should include instructions (either written on paper, or incorporated into the spreadsheet).

2. **Rugby Scores** (Careful and Accurate Programming)

Your programming work in this question will be assessed on two criteria:

(a) Completeness and accuracy of the program.

(b) Good presentation. That is, it should make good use of programming language facilities, be well organised, neatly laid out, and lightly commented.

The history of Rugby Union starts in 1823 when William Ellis is said to have picked up the ball and run with it, during a game of English School Football. The early game that developed from that event had a different scoring system to the modern version.

Note: It doesn't matter if you don't know what the events mean in the game. The only thing that is important is that a 'conversion' can only occur immediately following a 'try'.

The events that might lead to scores are

A "try" or "touch down":	In the modern game is worth 5 points In the original game gave no score (it just allowed players to 'try' for a conversion). Unconverted tries were counted and used for tie breaking in the event of a draw on points.
A conversion:	In the modern game is worth 2 points. In the original game is worth 1 point.
A penalty:	In the modern game is worth 3 points. In the original game is worth 1 point.
A dropped goal:	In the modern game is worth 3 points. In the original game is worth 1 point.

Note: The original game actually counted 'goals', but we call them 'points' here for simplicity.

Your task is to write a program to read a series of event descriptions and at the end of the game report the result as it would be in the modern game and as it would have been in the original game. Don't forget to count tries not immediately followed by conversions – so you can try to resolve ties in the original game scoring system.

We suggest that you build this as a 'console' or text interaction program. We do not expect any graphics or elaborate table display. Full marks can be achieved for a 'console' program.

Two sample interactions with a scoring program follow. User input is underlined. Note that you are not required to exactly match these programs.

In the first sample, the modern score is 15 to A and 6 to B. The original score is 2 to each team with A winning with one unconverted try.

```
Welcome to Rugby scoring

Instructions
. Refer to your teams as A and B
. Events are T(ry), C(onversion), P(enalty), or D(ropped goal)
. Enter events as "Event Team":  eg:  TA for a try by team A
. Enter X at the end of a game

Enter event: TA
Enter event: CA
Enter event: TA
Enter event: PA
Enter event: PB
Enter event: PB
Enter event: X
Modern Result: A wins
Original Result: A wins
```

In the second sample, the modern score is 15 to A and 9 to B. The original score is 2 to A and 3 to B; A’s unconverted try having no effect.

```
Welcome to Rugby scoring

Instructions
. Refer to your teams as A and B
. Events are T(ry), C(onversion), P(enalty), or D(ropped goal)
. Enter events as "Event Team":  eg:  TA for a try by team A
. Enter X at the end of a game

Enter event: TA
Enter event: CA
Enter event: TA
Enter event: PA
Enter event: PB
Enter event: PB
Enter event: PB
Enter event: X
Modern Result: A wins
Original Result: B wins
```

3. **STV** (Problem Solving and Programming)

Your programming work in this question will be assessed on two criteria:

(a) Your approach to the problem. We will be looking at your work for evidence that you found good ways of storing the necessary data, and devised algorithms for finding and displaying the requested results. ***Please hand in any notes and diagrams which describe what you are attempting to program, even if you don't have time to code or complete it. You may include comments in your program, or write a description of your program to hand in.***

(b) The extent to which your program works and correctly solves the problem.

You may find that the programming language you use makes it difficult to produce output as shown in the example implementation steps below. If this is the case, feel free to build your program in a way that suits your circumstances.

Different countries use different electoral systems when electing governments. In some systems, simple manual counting of votes is enough to work out a result. In others, more complex analysis of votes may be required. One system for which it is useful to have computer assistance is 'Single Transferable Vote'. In this question you are provided with voting results for an STV election. Your task is to analyse the votes and display results.

We will present the problem in stages for you to program. The stages are interleaved with explanation of aspects of the problem and some algorithm ideas. We suggest that you build your program in the order given. This will make it likely that you have parts working at the end, even if you don't have time to complete the whole program. However, we also strongly suggest that you read the whole problem statement before starting to program. We also suggest that you save working versions of your program at each stage.

Stage A

In an STV election, voters are allowed to put as many of the candidates as they wish in order of preference. The idea is that votes are counted for the first preference of each voter. The lowest polling candidate is eliminated from the voting. However, people who voted for the eliminated candidate have their votes transferred to their second preferred candidate (if any). In this way people who prefer an unpopular candidate can still have an influence the election outcome.

As an example, consider an election with five candidates, numbered 1 to 5. The votes cast by ten voters might be recorded like this:

```
2 1 4 5 3
2 1 5 3 0
1 2 3 5 4
4 5 2 3 1
5 4 1 2 0
1 4 2 5 0
4 1 5 2 3
4 1 2 5 3
2 5 4 0 0
4 1 3 2 0
```

Each row is the vote of one voter. The first voter (first row above) has ranked all five candidates, with candidate 2 as their first preference, 1 as their second, then 4, 5 and finally 3. The ninth voter (second to last line) has decided to rank only three candidates: numbers 2, 5 and 4.

Counting proceeds in steps. The first step is to count the first preference votes of each voter (for our example: the first column in the table above). Voters 1, 2 and 9 are counted as voting for candidate 2; etc. The result is:

```
First preferences
Candidate 1: 2 votes
Candidate 2: 3 votes
Candidate 3: 0 votes
Candidate 4: 4 votes
Candidate 5: 1 votes
```

Programming: As part of the examination material provided, you will have access to a file called STV.txt. It contains 1000 votes in an election with five candidates. The first stage of your program should read the file and count the first preference votes. They should be reported as shown above.

Stage B

The next step in the election counting is to find the candidate with the lowest number of votes in the first preferences. In the example, this is candidate 3, with zero votes.

Note: In real life we might have to deal with the possibility of two or more candidates having equal lowest scores. For this examination problem the data has been chosen to make sure that never happens.

The lowest polling candidate is then eliminated from the voting. This involves checking each vote one by one. If the lowest polling candidate is mentioned in the list of preferences, it is removed and the choices after it are shifted left by one place. Gaps at the right are filled with zeroes. For example the third voter’s options looked like this:

```
1 2 3 5 4
```

After elimination of candidate 3, they look like this:

```
1 2 5 4 0
```

Programming: Extend your program to find the candidate with the lowest number of first preference votes. Output a line reporting the elimination of that candidate and eliminate all votes for that candidate.

```
Candidate 3 eliminated
```

For our example the adjusted votes look like this:

```
2 1 4 5 0
2 1 5 0 0
1 2 5 4 0
4 5 2 1 0
5 4 1 2 0
1 4 2 5 0
4 1 5 2 0
4 1 2 5 0
2 5 4 0 0
4 1 2 0 0
```

Stage C

After elimination of a candidate, the first preference votes are counted again. In our example they haven't changed because there were no votes for candidate 3. However the list reported is shorter.

```
After first elimination
Candidate 1: 2 votes
Candidate 2: 3 votes
Candidate 4: 4 votes
Candidate 5: 1 votes
```

Again, the lowest polling candidate is eliminated. This time it is candidate 5.

```
Candidate 5 eliminated
```

After elimination the modified votes look like this:

```
2 1 4 0 0
2 1 0 0 0
1 2 4 0 0
4 2 1 0 0
4 1 2 0 0
1 4 2 0 0
4 1 2 0 0
4 1 2 0 0
2 4 0 0 0
4 1 2 0 0
```

Note that voter 5 is now voting for candidate 4 (their second preference) instead of 5.

Programming: Extend your program to recount and eliminate another candidate.

Stage D

The analysis continues finding and eliminating lowest polling candidates. In our example, after the fourth elimination, there is only one candidate left. Note that they don't end up with 10 votes as some are lost when voters don't rank all candidates.

```
After fourth elimination
Candidate 4: 9 votes
```

Programming: Complete the analysis, eliminating candidates and reporting the votes at each step, until only one remains. That candidate can be declared to be the winner. In our example:

```
Candidate 4 is elected
```

Stage E

Voting systems, like any other systems, have strengths and weaknesses. Often programmers, who look at the algorithms implementing systems in great detail and explore results obtained from different data sets, are in a good position to offer opinions on the merits or otherwise of those systems.

Describe a situation (pattern of voting) in which the STV result is probably not what most people would consider to be the fairest result. Can you think of an alternative way of analyzing the votes which would produce a better result? Note that this requires a written answer. You can use a text editor or word processor, or simply write your answer by hand.