mD2

35

THE UNIVERSITY OF THE WEST INDIES
ST. AUGUSTINE

EXAMINATIONS OF **December** 2017

Code and Name of Course: COMP1602 – Programming II

Date and Time: Thursday 21st December 2017 9am Duration: 2 Hours

INSTRUCTIONS TO CANDIDATES: This paper has 4 pages and 3 questions

Answer ALL Questions

Non-programmable calculators may be used.

Unless stated otherwise, assume that data given are valid.

1. (a) An m x n matrix is considered *sparse* if *at most* 20% of its elements are non-zero. A matrix `MM` is declared as `int MM[200][200]`. Write programming code to read, from a file *input.txt*, values for m and n, followed by mn integers to fill the matrix in *row-order*. Assume $m, n \leq 200$. Your code then calls the function *isSparse* (see next) and prints a message indicating whether or not the matrix is sparse. Include all relevant declarations.

Write a function *isSparse* which, given a matrix like `MM` and values for m and n, returns 1 if the matrix is sparse and 0, otherwise. [4]

(b) The *mode* of a list of numbers is the number which appears most frequently. For example, the mode of `3 7 8 3 7 5 3 2 3 9 8 6` is `3` since it appears the most times (`4`). A file *mode.in* contains numbers from `1` to `9` only. Each number can appear any number of times. There is no end-of-data marker but you can use the return value of *fscanf* to test when there is no more data.

Write a program to read the data and print the number of times each number appears. Also, determine and print the mode of the numbers. Ignore the possibility of a tie. Send all output to the file, *mode.out*. [6]

(c) The prime numbers from 1 to 400 can be obtained as follows. From a list of the numbers 1 to 400, cross out all multiples of 2 (but not 2 itself). Then, find the next number (`i`, say) that is not crossed out and cross out all multiples of `i` (but not including `i`). Repeat this last step provided that `i` has not exceeded 20 (the square root of 400). The numbers remaining in the list (except 1) are prime.

Write a program which uses this method to prompt for a value for `n` (`n` $\leq$ 999) and print all prime numbers from 1 to `n`, inclusive. The following is a sample run:

```
Find primes up to? 30
Primes from 1 to 30: 2 3 5 7 11 13 17 19 23 29
```

[5]

(d) An array `A` contains `n` integers that first increase in value and then decrease in value, for example, (`n` = 11)

17	24	31	39	44	49	53	29	20	18	13
0	1	2	3	4	5	6	7	8	9	10

It is unknown at which point the numbers start to decrease. Write *efficient* code to copy the numbers in `A` to another array `B` so that `B` is sorted in ascending order. Your code must take advantage of the way the numbers are arranged in `A`. Hint: perform a merge starting from both ends. [5]

Total Marks Q1: 20

2. (a) You are given the following function header:

```
int bSearch(int key, int list[], int lo, int hi)
//list is sorted in ascending order;
//search for key in list[lo..hi] using a binary search;
//if found, return its location; otherwise, return the location in
//which key should be inserted so that list would remain sorted
```

Write the function. [5]

(b) You are in charge of a men's high jump competition and you have data on each competitor, stored in a file *input.txt*. Each line of data consists of a jumper's number (integer), followed by his best 3 jumps (real numbers denoting height in metres). The number of jumpers is unknown but a line containing `0` only terminates the data. Here are some sample data:

```
5555 1.42 1.46 1.53
4590 1.51 1.56 1.49
3705 1.48 1.49 1.50
0
```

A jumper qualifies for the competition if the average of his 3 jumps is at least 1.50 metres. *Without using any array*, write a program to read the data and send the following to a file, *output.txt*.

For each jumper, print his number, his average jump and whether or not he qualifies. Print the total number of jumpers and the number that qualify. Print the jumper with the highest average jump (ignore the possibility of a tie). [7]

(c) You are given the same data as in part (b) but it is known that there are no more than 80 jumpers. Write a program to read the data and send a report to the file *output.txt* consisting of jumper number and average jump in descending order by average jump (that is, best jumper first). You *must* store the jumper number and average jump in an appropriate *array of struct*. Show all declarations. [8]

Total Marks Q2: 20

3. (a)

(i) What are pseudo-random numbers? [1]

(ii) Explain way it is necessary to "seed a random-number generator". [1]

(b) In the manufacture of electric bulbs, the probability that a bulb is defective is 0.01. Write a program to simulate the manufacture of 5000 bulbs, indicating how many are defective. Ensure that each time your program is run, a different answer could be obtained. [3]

(c) A barrel contains a large number of marbles of various colours. Marbles are picked at random from the barrel. There is a 50% chance of picking a red marble, a 20% chance of picking a blue marble, a 20% chance of picking a green marble, a 5% chance of picking a yellow marble and a 5% chance of picking a black marble. Assume that these percentages do not change after a marble is picked.

Write a program to perform 25 simulations of the picking of marbles until at least one marble of each colour is picked. For each simulation, print the number of marbles picked. Also print the average number of marbles picked per simulation. [7]

(d) State THREE differences between integers stored in a text file and integers stored in a binary file. [3]

(e) A retailer wishes to keep records of the items he sells. He has a text data file, `items.txt`, containing, on each line, an item number (integer), the quantity in stock (integer) and the unit price (real). The last line contains `0` only, indicating the end of the data.

Write a program to read the data (using `fscanf`) and store each item record in a binary file, `items.bin`. The records are to be stored in the same order as they appear in the text file. Your program should check if the input file could be opened and the output file could be created. [5]

Total Marks Q3: 20

Total marks on paper: 60

End of question paper