SPEC
228

THE UNIVERSITY OF THE WEST INDIES
ST. AUGUSTINE

EXAMINATIONS OF April/May 2018

Code and Name of Course: COMP 1602 – Computer Programming II

Date and Time: Wednesday 2nd May 2018 1pm Duration: 2 Hours

INSTRUCTIONS TO CANDIDATES: This paper has 5 pages and 3 questions

Answer ALL Questions

1) A C-string is a character array with a null terminating character (`'\0'`). In this question, you are *not* allowed to use any of the functions built into the C-string library.

   (a) Write a function, *append*, which accepts two C-strings ***s*** and ***t*** as parameters and adds the contents of ***t*** to the *end* of the contents of ***s***. Assume that ***s*** has enough space to hold the contents of ***t***. The prototype of *append* is as follows:

   ```
   void append (char s[], char t[]);
   ```

   [3 marks]

   (b) A C-string ***s*** contains a single integer value embedded among other characters, e.g., `"weight = 115 kg"`. Write a function, *getInt*, which accepts ***s*** as a parameter and returns the integer *value* stored in ***s***. Given the example C-string, *getInt* should return 115.

   [6 marks]

   (c) A *pangram* is a series of words which contains all the letters of the alphabet. The letters of the words may be in upper case or lower case. The most famous English pangram is probably:

   ```
   The Quick Brown Fox Jumps Over The Lazy Dog
   ```

   Write a function, *isPangram*, which accepts a C-string ***s*** as a parameter and returns *true* if the words in ***s*** form a pangram and *false* otherwise.

   [6 marks]

   (d) A C-string ***s*** contains a sequence of lowercase letters of the alphabet. Write a function, *longestSequence*, which accepts ***s*** as a parameter and returns the *length* of the *longest sequence* of *identical letters* in ***s***. For example, the function should return 6 if ***s*** is `"aabbcddddddeeebbbaaaacccd"`.

   [5 marks]

Total Marks: 20

2) A game is played on a board which is stored in a two dimensional (2D) array. The 2D array to represent the game board has a size of *numRows* and *numCols* where `0 ≤` *numRows* `< 100`, `0 ≤` *numCols* `< 100`, and *numRows* = *numCols*. A *Cell* struct stores the following data for **each** cell in the 2D array:

- Does the cell contain a light bulb?
- Is the cell hidden?
- The amount of immediate neighbours that contain a light bulb.

(a) Declare a struct, *Cell,* to store information on each cell and declare the array, *board*, to store the game board.

[2 marks]

(b) Write a function, *initBoard*, which initializes the array such that each cell:
  - o Does not contain a light bulb
  - o Is hidden
  - o Has zero neighbours which contain light bulbs

The prototype of *initBoard* is as follows:

```
void initBoard (Cell board[][100], int numRows);
```

[4 marks]

(c) Write a function, *isValid*, which given a particular location (row, column), determines if the location is valid for the given board. A location is valid if it is within the range of the rows and columns of the initialized board. The prototype of *isValid* is as follows:

```
bool isValid (Cell board[][100], int numRows, int row, int col);
```

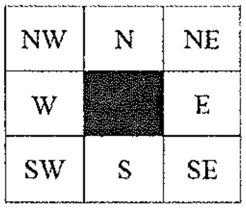
[3 marks]

(d) Write a function, *placeBulbs*, to randomly assign 10 percent of all the locations in the board with light bulbs. Note that every location on the board has two components, a row number and a column number. The prototype of *placeBulbs* is as follows:

```
void placeBulbs (Cell board[][100], int numRows);
```

[5 marks]

(e) A neighbour of a cell (shown in black on the diagram) is a cell that is North (N), South (S), East (E), West (W), North East (NE), South East (SE), South West (SW) or North West (NW) of the given cell, as shown in the diagram.



One or more neighbours may not exist if the cell in on the edge of the board.

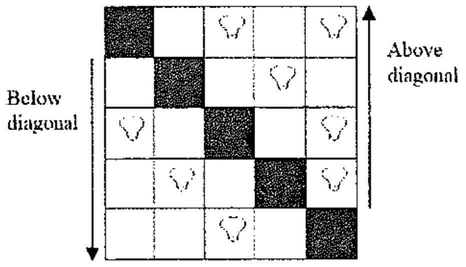
Write a function, *getNumber*, which given the 2D board and the location of a particular cell, finds the amount of neighbours of the cell which contain a light bulb. You can use functions previously written.

The prototype of *getNumber* is as follows:

```
int getNumber (Cell board[][100], int numRows, int row, int col);
```

[6 marks]

(f) The game board is said to be *crooked* if there are more light bulbs in the cells above the main diagonal than in the cells below the main diagonal. For example, the following game board is crooked:



Write a function, *isCrooked,* which determines if the game board is crooked. The prototype of *isCrooked* is as follows

```
bool isCrooked (Cell board[][100], int numRows);
```

[5 marks]

Total Marks: 25

3) (a) The array, *urban*, needs to be sorted in *ascending* order.

urban						
urban	Anon	Mila	Kamryn	Julia	Julia	Amara

Starting with the contents of *urban* as shown above, *draw* the modified array:

i. After each of the first three (3) passes of insertion sort;
ii. After each of the first three (3) passes of selection sort.

Show all working. [5 marks]

(b) The array below is sorted in *ascending* order:

12	17	35	47	89	99

Using a binary search technique, how many comparisons are needed to find the following keys?

i. 47
ii. 55

Show all working. [5 marks]

Total marks: 10

Total Marks: 55

End of Question Paper