JFK
169

THE UNIVERSITY OF THE WEST INDIES
ST. AUGUSTINE

EXAMINATIONS OF APRIL/MAY 2017

Code and Name of Course: COMP1602 Programming II Paper:

Date and Time: Monday 1st May 2017 4pm Duration: 2 Hours

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

Answer *all* questions.

Questions are not evenly weighted.

1. A C-string is a character array with a null terminating character ('\0'). Write code to implement the following functions involving C-strings. You are *not* allowed to use any of the functions built into the C-string library.

a. `int length(char s[])`

/* Returns the number of characters in `s`. */ [2]

b. `int add(char s[], char c)`

/* Inserts a character `c` at the end of `s` and returns the length of the C-string after the insertion.*/ [2]

c. `int lastOccur (char s[], char c)`

/* Searches for and returns the location of the last occurrence of the character `c` in `s`. If `c` is not in `s` return -1.*/ [2]

d. `int compareTo (char s1[], char s2[])`

/* Returns 0 if `s1` and `s2` are the same; -1 if `s1 < s2` and 1 if `s1 > s2`. */ [5]

e. A C-string, `haystack`, contains a special message. The message is enclosed by the tags '<' and '>'. Write a function which takes two C-strings, `haystack` and `message`, as parameters and finds and stores the message in `message`. The function has the following prototype:

`void getMessage (char haystack[], char message[])` [4]

Total marks: 15

2. A certain image, `picture`, consists of a set of pixels (picture elements) stored in a two-dimensional array of `100` rows by `150` columns.

Each pixel has a red value, a green value and a blue value which gives the pixel its colour. All three values are integers in the range 0 to 255.

a. Declare a struct of type `Pixel` to store a pixel's colour data. [1]

b. Write a function, `colour`, which accepts a `Pixel` variable and the red, green and blue values of the pixel as parameters and returns a `Pixel` variable with the given values. [2]

c. Declare a `Pixel` variable, `p` and assign the colour *orchid* to this variable.
(orchid: red = 218, green = 112, blue = 214) [2]

d. Declare `picture` as a two dimensional array of `Pixel` variables. [1]

e. Write a segment of code to colour all the pixels on the **top** and **right** edges of the image orchid. The top edge has a thickness of 4 pixels and the right edge has a thickness of 3 pixels. So, you must colour the four rows of pixels at the extreme top and the three rows of pixels at the extreme right of the image in orchid. [4]

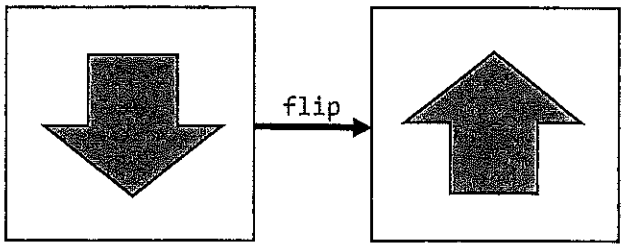
f. Assume that the data for an image has already been read from a file and stored in `picture`.

Write code to change the `Pixel` variables in `picture` to grayscale using the following procedure:

- For each pixel, use 30% of its red value, 59% of its green value and 11% of its blue value and set each colour of the pixel to this calculated value. [5]

g. The image in `picture` can be *flipped* by reversing the order of the rows, i.e.,
- Row 0 is switched with row 99
- Row 1 is switched with row 98
- Row 2 is switched with row 97
- And so on.

Write code to flip the image in `picture`.



[5]

h. If at least half of the pixels in an image is coloured black (red = 0, green = 0, blue = 0), it is referred to as a *sparse* image. Write code to determine if `picture` is sparse. [5]

Total marks: 25

3. Purchasing a Lotto ticket requires one to pick 7 numbers from the numbers 1 to 40. Write a program to randomly generate and print the numbers for 5 Lotto tickets, that is 5 sets of 7 numbers each, one set per line. No number is to be repeated in any of the tickets that is, exactly 35 of the 40 numbers must be used. If a number, say p, is generated which has been used already, the *first* unused number *after* p is used. Assume that 1 follows 40.
For example, if 15 is generated but has been used already, 16 is tried but if this has been used, 17 is tried and so on until an unused number is found. [10]

Total marks: 10

4. a. The array, num, needs to be sorted in *ascending* order.

num	32	8	29	6	17	32	3	12	8	23

Starting with the contents of num 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 bubble sort.
iii. If the array was already sorted, would insertion sort or bubble sort be more efficient?

Show all working. [6]

b. The array below is sorted in *descending* order.

34	32	29	23	17	12	8	6	3	1

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

i. 29
ii. 50

Show all working. [4]

Total marks: 10

End of Question Paper