University of Waterloo
CS240 - Spring 2017
Assignment 5

Due Date: Wednesday, July 19th, 5pm

Please refer to the course webpage for guidelines on submission. Submit your written solutions electronically as a PDF with file name `a05wp.pdf` using MarkUs. We will also accept individual question files named `a05q1w.pdf`, `a05q2w.pdf`, etc., if you wish to submit questions as you complete them.

Unless the base is explicitly specified, we assume in all questions that the logarithm operation is base 2.

Problem 1 Quadtrees [4 + 6 = 10 marks]

For all parts of this question, use the convention that each internal node of a quadtree has exactly four children, corresponding to regions NW, NE, SE and SW, in that order.

(a) One of the applications of quad trees is for image compression. An image (picture) is recursively divided into quadrants until the entire quadrant is only one colour. Using this rule, draw the quad tree of the following image in Figure 1. There are only three colors (shades of grey). For the leaves of the quad tree, use 1 to denote the lightest shade, 2 for the middle shade and 3 for the darkest shade of grey.

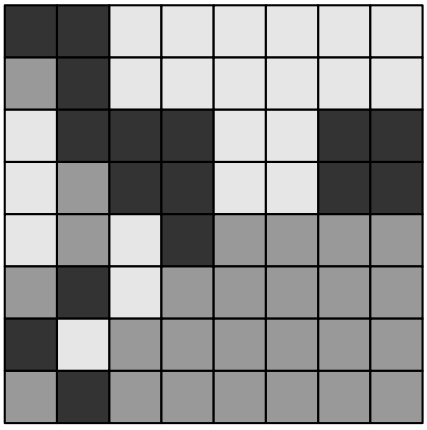

Figure 1: Grayscale Image

(b) Another application is to compare two images. Given two black and white images (i.e. each pixel of the image is either 0 or 1), each of size $2^k \times 2^k$ stored as quadtrees, give an algorithm to generate the intersection of the two images. If two corresponding pixels

are both 1, then their intersection is 1; otherwise their intersection is 0. No justification of correctness or runtime is required.

Problem 2 Range-Counting [6 + 6 (+5) = 12 (+5) marks]

(a) Assume that we have a set of n numbers (not necessarily integers) and we are interested only in the number of points that lie in a range rather than in reporting all of them. Describe how a 1-dimensional range tree (i.e., a balanced binary search tree) can be modified such that a range counting query can be performed in $O(\log n)$ time (independent of k, the number of nodes). Provide the range counting algorithm and justification of its runtime.

(b) Now consider the 2-dimensional-case, where we have a set of n 2-dimensional points. Given a query rectangle R, we want to find the number of points that lie in R. Preprocess the n points (by building an appropriate range-tree based data structure) such that you can answer any of these counting queries in time $O(\log^2 n)$. Provide the range counting algorithm and justification of its runtime.

(c) **(Bonus)** Suppose a two dimensional range tree data structure stores n points, and that the x-BST is perfect, i.e., every level is completely filled. Give an exact closed form formula in terms of n for the sum of the number of nodes in the x-BST plus the total number of nodes in all y-BSTs. No justification is required.

Problem 3 Pattern Matching [4 + 4 + 5 + 5 = 18 marks]

(a) Consider a pattern P and a text T. Assume that you are given the failure array for the string $P\Phi T$ (the concatenation of P, a character Φ that is not contained in P, and T). Explain how to use this array to find the first occurrence of P in T.

(b) Construct the last-occurrence function L and suffix skip array S for the pattern $P =$ `manannan` with alphabet $\Sigma = \{n, a, m, e\}$. Give your answers as tables as shown in Module 9.

(c) Show how to search for pattern $P =$ `manannan` in the text $T =$ `manaananenanmanannan` using the Boyer-Moore algorithm. Indicate in a table such as Table 1 which characters of P were compared with which characters of T. Follow the example on slides 32-33 in Module 9. Place each character of P in the column of the compared-to character of T. Put brackets around the character if they are known to match from the previous step. Use a new row when sliding the pattern. You may not need all space in the table.

(d) Let $M = 17$ be the prime number chosen by Rabin-Karp, $P = 181$ be the pattern to search for, and $h(k) = k \mod M$. Give a text T, with length 7, that produces the worst-case number of string comparisons for Rabin-Karp fingerprinting. Explain why T produces the worst-case.

m	a	n	a	a	n	a	n	e	n	a	n	m	a	n	a	n	n	a	n

Table 1: Table for Boyer-Moore problem.

Problem 4 *kd*-Tree Construction [10 + 10 = 20 marks]

(a) Implement an algorithm to construct a *kd*-tree for dimension 2. Your algorithm should read $2n+1$ integers from standard input, separated by white space. The first integer is the number of points. The remaining $2n$ integers are the points themselves, according to their x and y coordinates.

Actually, your program does not need to construct the tree, but rather should just print to standard output the n points in the order they are visited during a **pre-order traversal** of the *kd*-tree. Thus, your output should consist of $2n$ integers separated by whitespace.

- You may use any standard library function, for example, to sort an array.
- Your output must correspond to the tree produced by the recipe on Slide 13 of Module 8.
- Your answer to this question will be autotested. However, for the next part question regarding efficiency the graders will examine the code you have submitted.

Here is an example input and output file.

```
 Input:  4 3 4 2 2 0 1 1 3
Output:  2 2 1 3 0 1 3 4
```

(b) Analyze the (worst-case) running time of the algorithm you have implemented. The graders must be able to quickly match your analysis with the code you have submitted. If your analysis is correct you will receive for this question part:

- full marks for a linearithmic running time bound;
- otherwise, half marks for a subquadratic running time bound.
  (Note: An algorithm is said to run in linearithmic if its running time is $O(n \log n)$ and an algorithm is said to run in subquadratic time if its running time is $o(n^2)$.)