UNIVERSITY *of* LIMERICK

OLLSCOIL LUIMNIGH

FACULTY *of* SCIENCE *and* ENGINEERING

Department of Computer Science
and Information Systems

Final Assessment Paper

Academic Year:	2013/2014	Semester:	Autumn
Module Title:	Data Structures and Algorithms	Module Code:	CS4115
Duration of Exam:	2 hours	Percent of Semester Marks:	60
Lecturer:	P. Healy	Paper marked out of:	60

Instructions to Candidates:

- There are two sections to the paper: Short Questions and Long Questions
- The mark distribution is 15 marks for Short Questions and 45 marks for the Long Questions
- Answer all questions in all sections

Section 1. Short Questions (5×3 marks).

- Please put your answers to these questions in the answer book provided to you, labelling your answers 1.1, 1.2, etc.

1. The following code appeared in a `heapsort` implementation. Comment carefully on its purpose and operation.

```
for (int i = 0; i < n; i++)
{
  TYPE tmp = arr[i];
  arr[i] = arr[n-1-i];
  arr[n-1-i] = tmp;
}
```

2. The following code is a member function of a Binary Search Tree class. Explain what the function does and why.

```
template <class Comparable>
int
BST<Comparable>::whatEver( BNode<Comparable> *t )
{
  if( t == NULL )
    return 0;

  int l = whatEver(t->left);
  int r = whatEver(t->right);
  int m = max(l, r); // larger of two ints
  return m+1;
}
```

3. Give an example of a pair of functions

$f(n)$ and $g(n)$ where $f(n) = O(g(n))$ and $f(n) = \Theta(g(n))$.

4. Dijkstra's algorithm is one of the pre-eminent shortest path algorithms. Discuss how you might get a faster algorithm in the special case that all of the edges have identical weight.

5. We have seen that the running time of both Depth First Search and Breadth First Search are both $O(|V| + |E|)$. Comment on the space requirements of the two algorithms.

Section 2. Long Questions (45 marks).

- Please put your answers to these questions in the answer book provided to you
- Label your answers 2.1, 2.2, and 2.3 in your answer books

1. Fibonacci number generation. **(15 marks.)**

   (a) The Fibonacci number sequence is given by

   $$F_n = F_{n-1} + F_{n-2}, \quad n \geq 2, \quad F_0 = F_1 = 1$$

   Use induction to prove that $F_n \geq (\frac{3}{2})^n$. (3 marks.)

   (b) Write a recursive function `fib(n)` that computes the nth Fibonacci number. (3 marks.)

   (c) By drawing a tree of function calls argue that this function is not an efficient way to generate Fibonacci numbers. (3 marks.)

   (d) By counting the operations performed in your function relate the running time of your function to the Fibonacci numbers themselves and, hence, or otherwise, justify more formally your argument in the previous part. (3 marks.)

   (e) Write a much more efficient version of `fib(n)` to compute the nth Fibonacci number that runs in $O(n)$-time. What is the space requirement of your function? Can it be done using just constant space? Justify your answer. (3 marks.)

2. Sorting (and related) algorithms **(15 marks.)**

   (a) In the `quicksort()` algorithm we said that it is a bad idea to simply rely on the element in the first position of the array as a pivot when partitioning.

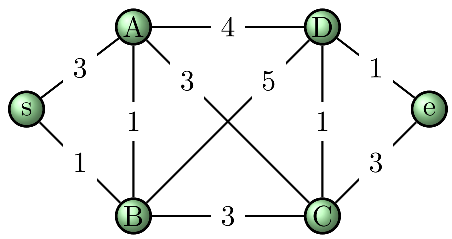
      i. Explain why this is. (3 marks.)

      ii. The usual solution to this problem is to use the *median-of-3* strategy to find an improved pivot. Give a convincing argument for why a more accurate pivot estimate from, say, a *median-of-5* algorithm has not become standard. (3 marks.)

   (b) Give an efficient algorithm, `kLargest()`, that, given an integer `k` and `arr[]`, an array of `n` numbers, returns the `k` largest elements of the array in `arr[n-k,...,n-1]`; that is, the last `k` positions of the array. The `k` elements do **not** have to be sorted. What is the running time of your algorithm? (9 marks.)

3. Graph Algorithms. **(15 marks.)**

(a) For the graph shown below describe in detail the steps taken by Dijkstra's algorithm when computing the shortest path from s to e. A matrix along the lines of what we used in class will be a good way of showing your calculations / steps. (9 marks.)



(b) Given two columns of data, comprising currency names and exchange rate (expressed as, say, 1.34USD= 1€) explain how you could use a graph and a path-length graph algorithm we have studied to make money fast. That is, is there a sequence of exchanges that could be found that would result in instant profit? For instance, if the currencies are x, y and z, and the exchange rate is $1x = 2y$, $1y = 2z$ and $1x = 3z$ then $300z$ will buy $100x$, which in turn will buy $200y$ and then $400z$. So $300z$ will buy $400z$ and a profit of $100z/300z = 33.3\%$ results. (6 marks.)