



# UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

FACULTY of SCIENCE and ENGINEERING

Department of Computer Science  
and Information Systems

## Final Assessment Paper

Academic Year:	2012/2013	Semester:	Spring
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 20 marks for Short Questions and 40 marks for the Long Questions
- Answer all questions in all sections

### Section 1. Short Questions ( $5 \times 4$ 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 height of an AVL tree is no worse than \_\_\_\_\_ times the optimal height.
2. Give the recurrence relation for  $N_h$ , the number of nodes in the worst possible AVL tree of height  $h$  \_\_\_\_\_?
3. In a  $d$ -heap (a heap where each node can have at most  $d$  children), what are the locations of a node's children? The root node of the heap is at location 1. \_\_\_\_\_
4. Sorting is possible in  $o(n \log n)$ -time with \_\_\_\_\_ sort.
5. If a graph has negative edge costs then the "Big-Oh" running time of the shortest path algorithm increases to \_\_\_\_\_.

## Section 2. Long Questions (40 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. Prime number detection.

(15 marks.)

- Write an efficient function (in C++ or pseudo-code) to check if a given number is prime. What is the running time of your program? Your solution should differ from that given in the next part of the question since it asks something different. (5 marks.)
- The code below does something different to the previous task. Called the *Sieve of Erasthenes*, it finds *all* prime numbers up to some given value. Explain *in detail* why it works. (5 marks.)
- Analyse its running time and give as good an estimate of its worst-case behaviour as you can. (5 marks.)

```
void genPrimes(int n)
{
    //create prime array
    bool *prime = new bool[n+1];

    //set 0 and 1 as not prime
    prime[0] = prime[1] = false;

    //set remainder 'true' initially
    for (int ind = 2; ind <= n; ind++)
        prime[ind] = true;

    //elimination time
    for (int p = 2; p <= (int)sqrt(n)+1; p++)
    {
        if(prime[p] == false) continue;

        for (int c = 2*p; c <= n+1; c = c+p)
            prime[c] = false; // multiple of p so composite
    }
}
```

### 2. Linked Lists.

(15 marks.)

Suppose you have two polynomials  $P(x)$  and  $Q(x)$  which are to be added together. Assume that  $P(x)$  has  $n$  non-zero terms and that  $Q(x)$  has  $m$  non-zero terms, where  $n > m$ . The polynomials are represented using linked lists for reasons of space efficiency and the lists should store the non-zero terms ordered from largest power to smallest power.

- Draw a picture of the linked-list representation of  $P(x) = -2x^{16} + 4x - 3$ . (3 marks.)
- Give an  $O(n)$ -time algorithm for performing the addition. Your algorithm should be written in C++ or, failing that, pseudo-code. You may assume that your `Term` class has appropriate constructors. (9 marks.)
- Is linear-time the best possible time asymptotically for part (b)? Why? (3 marks.)

3. Graph Algorithms.

**(10 marks.)**

The Erdős Numbering Problem is to determine for a given computer scientist how *closely related* the researcher is to Paul Erdős, often considered to be one of the greatest mathematicians / computer scientists ever.

The Erdős Number (EN) of Erdős himself is 0; the Erdős number of an author is defined to be 1 if the author has co-written a paper in a mathematical journal with Erdős; otherwise, to determine an author's EN, find the lowest EN amongst all the people he has ever co-written a paper with and add 1 to that.

So, if author "Joe Bloggs" has EN 7, then "Seoirse de Blogg", a colleague and co-author of Joe Bloggs', will have EN *at most* 8. Note that his EN could possibly be lower than 8 if he has worked with somebody else who is more closely related to Erdős.

Using data structures and algorithms that we have studied in class develop an algorithm that reads a database of co-authors and determines the EN of some specified author.

What is the worst-case running time of your algorithm? What are its space requirements?