2009 Paper 4 Question 1

Algorithms II

(*a*) Explain the terms *amortized analysis* and *aggregate analysis*, highlighting the difference between them. [3 marks]

(*b*) Assume that the following two classes, which implement the standard stack (last-in first-out) and queue (first-in first-out) data structures, are available.

```
class Stack                          class Queue
    void push(Item x)                    void enqueue(Item x)
    Item pop()                           Item dequeue()
    Boolean isEmpty()                    Boolean isEmpty()
```

(*i*) A `Multistack` class is derived from `Stack` with the addition of two methods:

- a `void multipush(Itemlist l)` that takes a list `l` of items and pushes each of the items onto the stack (each action of extracting an item from the list and pushing it onto the stack has constant cost), and

- a `void multipop(int m)` that takes an integer `m` and pops that many items off the stack (raising an exception if there were fewer, but don't worry about that).

Is it true or false that, given an arbitrary sequence of n `Multistack` operations starting from an empty `Multistack`, each operation in the sequence has amortized constant cost? Justify your answer in detail. [5 marks]

(*ii*) Provide an implementation of class `Queue` using no other data structures than `Item`, `Boolean`, `int` and `Stack`. The amortized running time of each `Queue` method must be constant. (*Note that you may only use the* `Stack` *as a black box: you are not allowed to access its internal implementation.*) [7 marks]

(*iii*) Using the potential method, prove that the amortized running time of all your `Queue` methods from part (*b*)(*ii*) is indeed constant. [5 marks]