6 Programming in C and C++ (djg11)

(*a*) A C programmer has an array of pointers to structs. They sort the array using the built-in comparison operator, '<'. Describe one circumstance where this will produce undefined behaviour and another situation where the behaviour is defined. [2 marks]

(*b*) Give three reasons why is it helpful to separate the declaration and implementation of classes in OO programming. The following code gives a (possibly incorrect) C++ signature declaration for a collection type. Criticise this definition and explain any difficulties that might arise with having the implementation in a separate file. If all types entered in the collection inherit from a common parent, can this make a difference? [8 marks]

```
template <typename T> class bucket
{ public:
  bucket(int N);     // Constructor: holds up to N items.
  int push(T *item); // Add item or return non zero if full.
  T pop();           // LIFO order pop most-recently added.
  T *dequeue();      // FIFO order dequeue of oldest item.
};
```

(*c*) Instead of holding pointers to objects, a collection class can hold the objects themselves. What changes in the `bucket` method types would be needed? What are the advantages and disadvantages of this change? [4 marks]

(*d*) Returning to holding references in the collection, it is instead required that all objects pushed must have a `void foo()` method that the `bucket` will invoke for all members currently in the `bucket` on either removal operation. Define how items might be stored inside the `bucket` and sketch suitable code for the `dequeue()` method with this addition. Can the `foo()` method be invoked using static or dynamic method invocation? [6 marks]