Question 1 (18 marks)

A double-ended queue (or "deque") is a queue that permits enqueues and dequeues to be performed at either end. The basic operations are "enequeue at front", "enqueue at end", "dequeue from front", and "dequeue from end". Dequeues can be very conveniently implemented using a doubly linked list.

a) Your answer booklet contains most of the class declaration for a deque of double values. You are to fill in the missing private portion. Be sure to indicate (with appropriate comments) which end of the doubly linked list is the "front" of the deque.

b) Give the implementation of the destructor.

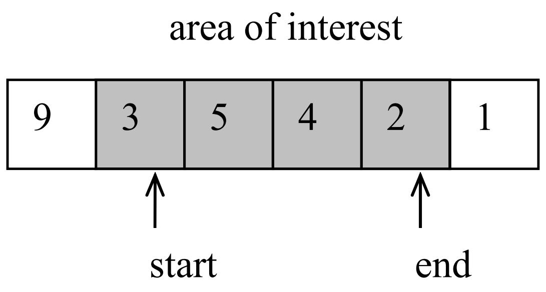
c) Give the implementation of "dequeueAtEnd".

d) Give the implementation of method "size".

e) Give the implementation of the copy constructor.

Question 2 (6 marks)

In the space provided in your answer booklet, write a **recursive** function that finds and returns the largest value in some portion of an **unsorted** array. The area of interest starts at element "start" and runs through element through "end" (as illustrated below).



A function header has been supplied (see your answer booklet). You should definitely not need all of the space provided for your answer.

You should work on the assumption that "start" will never be greater than "end".

Question 3 (12 marks)

This question assumes the class outlined below. Note that only those aspects of the class that are relevant to the question have been shown. Other aspects have been omitted.

```
class StudentDB {
private:

    class TNode {
    public:
        int number; // student number
        double GPA;
        TNode *left, *right;
    };

    TNode *head;
    int count;  // total number of students

public:

    // returns the number of students with a GPA of 10.0 or more
    int countGoodStudents () const;

    // inserts a new student into the tree or,
    // if the student already exists, updates the student's GPA
    void insertStudent (int number, double GPA);

};
```

The class involves a tree. This tree is sorted by student number in the usual manner (i.e. the lowest student number is at the very left of the tree).

In the space provided in your answer booklet, implement methods "countGoodStudents" and "insertStudent". You may use whatever techniques you think are appropriate. You may use recursion if you want to but do not have to use it. You may assume any number of private member functions as long as you implement these functions as well.

Note that the class includes a "count" variable.

Question 4 (6 marks)

This question assumes the stack and queue classes outlined below. Note that these class definitions are not complete. Instead you have only been given what you need to answer this question.

```
template <class T>
class Stack {
public:

  Stack(); // creates an elastic stack

  push (const T &data);
  T pop (); // throws an "overflow_error" exception if the stack is empty
  int size (); // returns the number of values on the stack
  bool isEmpty(); // returns true if the stack is empty

};


template <class T>
class Queue {
public:

  Queue(); // creates an elastic queue

  enqueue (const T &data);
  T dequeue (); // throws an "overflow_error" exception if the queue is empty
  int size (); // returns the number of values in the queue
  bool isEmpty(); // returns true if the queue is empty

};
```

Write a function that, given a queue of integer values, reverses the order of the queue's contents. Your function must conform to the sample call below.

```
Queue<int> queue;

// assume that some values get enqueued here …..

reverseQueue (queue); // reverse the order of the values on the queue

// the value that was at the head of the queue is now at the end of the queue, and so on
```

Question 5 (18 marks)

A web browser intended for use by children maintains a list of forbidden words. If a web site contains words that are on the list, the web site is blocked and cannot be viewed. As the list of forbidden words might be quite long, it is essential that it be possibly to quickly determine whether or not a word is on the list (just in case it isn't obvious, this is a hashing question).

a) Your answer booklet contains most of a suitable class declaration. Fill in the missing private portion. You may choose to use either linear rehashing or hashing with buckets. Indicate which technique you think you're using.

b) Give the implementation of the constructor.

c) Give the implementation of "addWord".