Question 1 (a): Complete the class declaration.

```
class Deque {
private:











public:

   Deque ();
   ~Deque();
   Deque (const Deque &otherDeque)

   enqueueAtFront (double value);
   enqueueAtEnd (double value);

   // both of these methods throw an overflow_error exception if the deque is empty
   double dequeueFromFront ();
   double dequeueFromEnd ();

   int size(); // returns number of names in the deque

   Deque &operator= (const Deque &otherDeque);

 };
```

Question 1 (b): Implement the destructor here.

```
Deque::~Deque () {

}
```

}

Question 1 (c): Implement method "dequeueFromEnd" here.

```
string Deque::dequeueFromEnd () {

    ________________________________________________

    if (size() == 0) throw overflow_error ("Deque is empty");

    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________
    ________________________________________________

}
```

Question 1 (d): Implement method "size" here.

```
int Deque::size () {

    ________________________________________________
    ________________________________________________

}
```

Question 1 (e): Implement the copy constructor here.

```
Deque::Deque  (const Deque &otherDeque) {

}
```

}

Question 2: Write your function here.

```
int findLargest (int array[], int start, int end) {

}
```

}

Question 3: Write the implementation of "countGoodStudents" here.

int studentDB::countGoodStudents () const {

}

}

Question 3: Write the implementation of "insertStudent" here.

```
void StudentDB::insertStudent (int number, double GPA) {

}
```

Question 4: Write your function here.

Question 5: Complete the following class definition.

```
class WordList {
private:

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   ________________________________________________

   int hash (const string &word, int n); // returns a value between 0 and n – 1 (inclusive)
   void standardize (string &word); // converts word to all uppercase

   // declared private in order to make these operations illegal
   WordList (const WordList &otherList);
   WordList &operator = (const WordList &otherList);

public:

   WordList (int maxWords);
   ~WordList ();

   addWord (const string word); // adds a new word to the list
   bool lookupWord() const; // returns true if the word is in the list, false otherwise

};
```

Question 5 (b): Implement the constructor here.

```
WordList::WordList (int maxWords) {

}
```

Question 5 (c): Implement method “addWord” here

WordList::addWord (const string word) {

}