Study Guide:

You are responsible for all reading, all worksheets, and all homework assignments posted on the course webpage. The worksheets and homework are an excellent study source. Make sure that you understand the solutions that we have sent you via the class e-mail. If you lost some points on homework, check where and why you made the mistake.

To prepare well for the midterm exam, make sure you are comfortable with all data structures, and all implementations of the data structures that we have discussed. For example, it would not be sufficient to only know how to implement a deque using the dynamic array. You also need to know how to implement a deque using singly-, doubly, and circularly-linked lists.

You should be able to define the interface and implement: Bags, Stacks, Queues, Deques, and Sets using dynamic arrays, singly linked lists, doubly linked lists, and circularly linked lists.

You should be familiar with the big-O complexity of all the functions from the worksheets and homeworks. You should also know how to compare various implementations that we have discussed in terms of the big-O complexity.

You will be required to write some code. This will be similar to the worksheets and homework, i.e., we will provide some functions that you would need to complete. Below, we list a few example problems for your practice.

1. What is the big – O complexity for:
   a. A method that takes exactly 2n²+5n+100 steps
   b. for (i = n; i > 0; i = i/2) {
       //constant time operations
   }
   c. for (i = n; i > 0; i = i/2) {
      for (j = i; j > 0; j = j/2) {
        //constant time operations
      }
   }
   d. for (current = previous->next; current != 0; current = current->next) {
      for (previous = current->next; previous != 0; previous = previous->next) {
        //constant time operations
      }
   }
2. Write elements of the stack, implemented using a dynamic array, as well as the size and capacity of the stack, after the following commands are executed.

```c
Struct dynArr s;
initDynArr(&s, 4);
pushDynArr(&s, 6);
pushDynArr(&s, 3);
popDynArr(&s);
pushDynArr(&s, 10);
pushDynArr(&s, 15);
pushDynArr(&s, 2);
pushDynArr(&s, 20);
popDynArr(&s);
```

9. Write the indexing function TYPE getValueAt() for the doubly-linked list implementation of a deque that takes an integer index, and returns the value stored at that index assuming index 0 is the link after the front sentinel. What is the big-O execution time of the function that you provided?

```c
struct dlink {
    TYPE value;
    struct dlink * next;
    struct dlink * prev;
};
struct listDeque {
    int size;
    struct dlink *frontSentinel;
    struct dlink *backSentinel;
};

TYPE getValueAt(struct listDeque *q, int idx){
    // FIX ME
}
```