1) remove() has time complexity O(1) since it only needs to perform a single operation everytime. Regardless of how many elements are in the list, a constant amount of operations take place. getValue(index i) will also have a time complexity O(1), for the same reason. There are no loops, or recursion, or searching of any type which would introduce more complexity. Remove composed only of the three highlighted operations, and getValue is composed of two highlighted operations.

**public** **int** remove(){

**int** val = queue[--N];

System.***out***.println(val);

**return** val;

}

**public** **int** getValue(**int** i){

if(i>=N) return -1;

return queue[i];

}

2) The add function as programmed will have a run time O(N) because every time it is called, it will shift the whole array to the right by one element. Performing this shift has complexity O(N) since it will have to iterate from N to 0 every time in the for loop. All other operations are constant time.

**public** **void** add(**int** a){

N++;

shiftRight();

queue[0] = a;

}

**private** **void** shiftRight(){

**for**(**int** k = N; k >0; k--){

queue[k] = queue[k-1];

}

}