Lab02: Arrays and Linked Lists

Note: The codes are given in C++, but you can choose to implement equivalent ones in Java.

Ex1.

Consider the following definition for an array. Note that you have to devise of a way to test all of your implemented functions.

```
static const int MAX = 500;
int array[MAX];
int length = 0;         //The size of the array
```

1. Write a function to create a new array from a file where all space-separated numbers are shown on the first line:

```
int[] newArray(char *filename, int &length){
}
```

2. Write the implementation for the following functions:

```
//Print out the elements in the array
void printArray(int array[], int length){
}

//Insert an integer value to the array at index^th
//position. The first element has index 0. Returned
//result is the array after the insertion.
//Do nothing if the index is out of range.
int[] insertNum(int array[], int &length, int value,
    int index){
}

//Remove an element at index^th position of the array.
//Returned result is the array after the removal.
//Do nothing if the index is out of range.
int[] removeIndex(int array[], int &length, int index){
}
```

```
//Find the first index of a given element in the array
// -1 if value is not in the array
int findIndex(int array[], int length, int value){
}
```

Ex2.

Consider the following definition for a linked list. Note that you have to devise of a way to test all of your implemented functions.

```
typedef struct Node ListNode;
struct Node{
   int data;
   ListNode *next;
}

typedef struct FirstNode *LinkedList;
struct FirstNode{
   ListNode *first;
}
```

1. Write a function to create a new list from a file where all space-separated numbers are shown on the first line:

```
LinkedList newList(char *filename){
}
```

2. Write a function to create a new node containing the value v:

```
// Create a new node containing a given number
ListNode newListNode(int v){
}
```

3. Write the implementation for the following functions:

```
//print out the elements in the list
void printList(LinkedList l){
}

//add a new number to the beginning of the list
void addFirst(LinkedList l, int v){
}

//add a new number to the end of the list
void addLast(LinkedList l, int v){
}

//remove the first element in the list
void removeFirst(LinkedList l){
}

//Find the first index of a given element in the list.
//The first element has index 0.
// -1 if v is not in the list
int findIndex(LinkedList l, int v){
}
```