# Lab #9

CS-2050 - Section D

Week of April 5, 2021

## 1 Requirements

In this lab, you will write a set of functions for maintaining a **linked list**. The structure of the list is *not clearly defined*, and you must design the list structure to meet the performance constraints outlined in the functions below. There are at least two list designs that could be used to achieve these performance constraints, which you have learned about in lecture.

Your implementation of the required functions should match the performance requirement *exactly*. The structure of your list should reflect the expected performance of each function, and no functions should perform **faster or slower** than expected.

```
> lab.h
typedef struct List List;

> lab.c
struct List {
    // What goes here is up to you
};
```

**Info:** Unlike with previous labs, you will be graded in this lab for your implementation of the List structure itself. You may choose to include as many or as few List struct members as you like, and you may define additional struct types to complete this assignment. The names of struct types and members should be **clear and accurate** as to their purpose, and you should **leave comments** describing the structure of your list implementation.

### 1.1 Support Functions

```
// COMPLEXITY: O(1)
List* initList();
// COMPLEXITY: O(1)
int getSize(List *list);
// COMPLEXITY: O(n)
void* getAtIndex(List *list, int index);
// COMPLEXITY: O(n)
void freeList(List *list);
```

**Info:** These functions are required for grading purposes, but are not part of the testing for this lab. You are expected to implement these functions to support your list implementation, but they are being counted as a single "function group" and will not be a significant part of your grade for this lab.

### 1.2 insertAtTail

```
// COMPLEXITY: O(1)
int insertAtTail(List *list, void *object);
```

**Info:** Inserts the given object at the *tail of the list* in **O(1) time**. Returns 1 on success, else 0.

### 1.3 getHead

```
// COMPLEXITY: O(n)
void* getHead(List *list);
```

**Info:** This function returns the object at the *head of the list* in **O(n) time**

### 1.4 removeHead

```
// COMPLEXITY: O(n)
void* removeHead(List *list);
```

**Info:** This function removes and returns the object at the *head of the list* in **O(n) time**

## 2 Notice

**Grading: Total 32 points**

1. Write required *support* functions
   * 8 points
2. Write required *insert* function
   * 6 points
3. Write required *get* function
   * 6 points
4. Write required *remove* function
   * 6 points
5. List structure is properly formatted and supports performance requirements
   * 6 points

**Notice:**

1. All of your lab submissions must compile under GCC using the $-Wall$ and $-Werror$ flags to be considered for a grade.
2. You are expected to provide proper documentation in every lab submission, in the form of code comments. For an example of proper lab documentation and a clear description of our expectations, see the lab policy document.