# 1 Objectives

In addition to practicing the writing of C programs, the objectives of this lab are to get you somewhat familiar with:

1. Reading code written by someone else.
2. Using `valgrind` to detect and locate memory errors.
3. Linked lists.
4. Formatting output with `printf`.

# 2 Requirements

**Big picture**: you are given a partially completed C program that uses linked lists. The first task is to finish the incomplete parts, while the second task is to turn in a completed program with no memory-related problems.

## 2.1 Program Description

This program generates a list of random numbers that are stored via a linked list (one object per number). The size of the list is also determined randomly, up to a maximum of 19 numbers. Each number is stored in a "val" field of the structure. After the list is generated, a function is called to "walk" through the linked list, adding up the numbers as it goes. After the totals have been calculated, the program then calls a function to display the random numbers. For example, the output shall look like the following:

```
Length of list = 12

Number  1 = 26
Number  2 = 17
Number  3 = 10
Number  4 = 17
Number  5 = 13
Number  6 = 34
Number  7 =  4
Number  8 = 26
Number  9 = 41
Number 10 = 33
Number 11 =  1
Number 12 = 20

Total = 242
```

The displayed numbers shall be **right-justified** as shown in the example to allow for a 2-digit number.

### 2.2 Task 1 – Complete the Program

1. The `build_list()` function is already provided for building the linked list.
2. You shall complete the `calc_total()` and `print_list()` functions as described above. These functions are already declared in the provided `randadd1.c` file, with some additional description in the comments.
3. You shall implement the right-justified output using a formatting option that is available with `printf()`.
4. Do not change the inputs and outputs of the functions.
5. Do not add global variables when modifying the code (for either task).
6. Do not add any `exit` calls in the code. Errors shall be detected by the called function and returned to the caller, and then finally returned by `main`.
7. Detected errors should display meaningful messages to the user.

### 2.3 Task 2 – Find and Correct Memory Errors

1. The provided code has some intentionally inserted memory-handling errors that need to be found and fixed. In addition, your own code should not introduce additional errors.
2. Use `valgrind` to help with this task. When your instructor runs `valgrind` on your code, it should not complain about memory problems. The following `valgrind` options may be useful: `--leak-check=full, --verbose, --help`. Don't forget to compile with the `-g` option to help you identify the offending lines of code.
3. Fix the problems with the minimal amount of change to the code. For example, do not totally rewrite `build_list()` if there is a problem there. Instead, fix the code that has a problem.
4. You are not to share the problems that need to be fixed in the code; every student needs to find the issues independently.
5. Add an entry at the bottom of the file header to summarize the changes you made to the file.

## 3 Submission

Submit `randadd1.c` to Sakai by the deadline.

## 4 Grading

Severe deductions shall be given in the following situations:

1. Compilation errors (i.e., your code will not compile). (-30 points)
2. The program crashes, such as a segmentation fault. (-20 points)
3. Only some of the requested functionality is provided.
4. I reserve the right to include other reasons I have not yet encountered.

The following will result in point deductions of various degrees:

1. The program generates warnings when compiled with the "-Wall" option. (-10 points)
2. Print statements meant for debugging were not removed or commented out (so they are part of the output). (-10 points)
3. Deviations from the style guide. (-1 point for each violation)

4. Incorrect output is provided (such as an incorrect total). (-10 points)
5. Incorrectly formatted output is provided (i.e., it does not comply with the specification given above). (-2 points)
6. Valgrind reports memory problems. (-10 points)
7. I reserve the right to include other reasons.