# CS0011 Introduction to Computing for Scientists
# Project 2
# Lab 2

## Lab 2

### Part 1: Representing the data as a list of lists

For this lab we will be working with 3 files:

ALL_AML_training.tsv
ALL_AML_testing.tsv
ALL_AML_labels.tsv

You worked with `data_set_ALL_AML_train.txt` and `table_ALL_AML_samples.txt` in Lab 1. As part of that analysis, you were able to see that these files were not properly TSV formatted. Fortunately, your instructor has cleaned up these files to create `ALL_AML_training.tsv` and `ALL_AML_labels.tsv` that are properly TSV formatted. Additionally, you will be working with `ALL_AML_testing.tsv`.

These files present the gene expression data from different samples taken from ALL and AML leukemias. `ALL_AML_labels.tsv` lists each sample ID number and identifies whether each sample was diagnosed as ALL or AML. The gene expression measurements for each of these samples are split across `ALL_AML_training.tsv` (samples 1-38) and `ALL_AML_testing.tsv` (samples 39-72). Each line of `ALL_AML_training.tsv` and `ALL_AML_testing.tsv` lists the expression data of a given gene. The first item on each line (before the first tab) describes the gene. The second item on each line (between the first and second tabs) gives the gene accession number. The remainder of each line will least the gene expression measurements for each of the samples contain within that file. Note that these "columns" are not in order.

Each line of `ALL_AML_training.tsv` lists the sample readings in the following order:

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
24 25 26 27 34 35 36 37 38 28 29 30 31 32 33
```

Each line of `ALL_AML_testing.tsv` lists the sample readings in the following order:

```
39 40 42 47 48 49 41 43 44 45 46 70 71 72 68 69 67 55 56 59
52 53 51 50 54 57 58 60 61 65 66 63 64 62
```

The first thing that you'll do is to write two functions named `read_data_tsv` and `read_label_tsv`. Both of these functions should:

- Take a single string argument, the filename of the TSV file that should be read.
- Return a nested list data structure that is a list of lists. Each sublist should contain all of the items from a line of the file. Make sure to convert any integer in the file to an `int` before adding it to a sublist.

You can use `read_data_tsv` to process `ALL_AML_training.tsv` and `ALL_AML_testing.tsv`. You can use `read_label_tsv` to process `ALL_AML_labels.tsv`.

After writing these functions, you should rewrite your `main` function to use these functions to generated nested lists storing the data found in `ALL_AML_training.tsv`, `ALL_AML_testing.tsv`, and `ALL_AML_labels.tsv`. Have your `main` function print out each of these list of lists to the terminal to ensure that they are working properly. Specifically, your main function should:

1. Assign the variable name `labels` to the result of calling `read_label_tsv` for `ALL_AML_labels.tsv`
2. Assign the variable name `training_data` to the result of calling `read_data_tsv` for `ALL_AML_training.tsv`
3. Assign the variable name `testing_data` to the result of calling `read_data_tsv` for `ALL_AML_testing.tsv`.
4. Print `testing_data` to the terminal
5. Print `training_data` to the terminal
6. Print `lables` to the terminal

## Part 2: Calculating averages

Next, you should write a function named `calc_avgs` that accepts 2 arguments. The first should be a list of lists containing the diagnosis of each sample (i.e., the nested lists representing the contents of `ALL_AML_labels.tsv`). Name this argument `labels`. The second should be a list of lists containing the gene expression measurements of a set of samples (i.e., the nested lists representing either the contents of `ALL_AML_training.tsv` or `ALL_AML_testing.tsv`). Name this argument `data`.

This function should, for each gene, print out the mean expression value for ALL samples and the mean expression value for AML samples. Note that this will produce thousands of lines of output when processing the provided files.

For each "row" of the data, you will need to determine the gene represented by that row, and then, for each "column" of that row, determine if the value at that index corresponds to an ALL or AML sample using the sample ID for that column (found in the first row), and the data in `labels`.

Once you have this function working, you should have your main function call it with `labels` and `training_data` as arguments. The last few lines of output produced should be:

```
... THOUSANDS OF PREVIOUS LINES OUTPUT ...

U73738_at ALL mean: 8.148
U73738_at AML mean: 10.455
```

```
X06956_at ALL mean: 420.852
X06956_at AML mean: 345.818

X16699_at ALL mean: -21.000
X16699_at AML mean: -17.727

X83863_at ALL mean: 769.222
X83863_at AML mean: 1114.091

Z17240_at ALL mean: 340.185
Z17240_at AML mean: 325.182

L49218_f_at ALL mean: 23.519
L49218_f_at AML mean: 8.636

M71243_f_at ALL mean: 243.593
M71243_f_at AML mean: 1144.545

Z78285_f_at ALL mean: -31.259
Z78285_f_at AML mean: -24.182
```

As you can see, some genes have drastically different expressions in ALL vs AML. We will use this to try to help identify ALL vs AML in Lab 3.

Once you have completed all parts of the lab, be sure to show your work to the lab instructor.