

Laboratory Assignment #2— Command line arguments and malloc

Due: at 5:00 pm on 3/30

Outcomes

After successfully completing this assignment, you should be able to:

- Take command line arguments for your program

Background Information

So far in class, we have primarily used `printf()` and `scanf()` to get information from the user. But what if we want to read a lot of data? And wouldn't it be easier if instead of waiting for the code prompt you for the size of your data, you could just type

```
./a.out 8
```

when you run your code (with 8 being the size)? In this lab, we will use `argc` and `argv` to get command line arguments and use them to read data from a file.

Set up

Download all files from Files-> Labs -> Lab 3.

Read `calc_avg.c` and make sure you understand how the program works. Compile and run `calc_avg.c`.

Command line arguments:

Command line arguments allow us to pass information to our program without waiting for a prompt. We have already been doing this every time we compile our code:

```
gcc -Wall calc_avg.c
```

In this example, the program we're running is `gcc`, and the arguments are `-Wall` and `calc_avg.c`.

To get command line arguments in our program, we need to modify the declaration of `main`:

```
int main(int argc, char *argv[])
```

`argc`: the number of command line arguments

`*argv[]` a pointer to an array of strings that contains the command line arguments.

For example, if we call

```
./a.out 5 7
```

`argc` will equal 3 (the first argument is the name of the program, i.e. `a.out`)

`argv[0]` will equal "a.out"

`argv[1]` will equal "5"

`argv[2]` will equal "7"

You'll notice all of the values are in quotes, that is because they are read in as strings. It is your job to parse them into numbers. To convert to an integer, use the function `atoi()`;

```
int nRows = atoi(argv[1]);
```

```
int nCol = atoi(argv[2]);
```

When using command line arguments, it is important to add a statement in your README file to explain the arguments that should be passed when running the program. For example:

To run this program, type `./a.out nRows nCol`

where `nRows` is the number of rows and `nCol` is the number of columns.

You can read more about command line arguments at
https://www.montana.edu/rmaher/ee475_fl04/Command_line.pdf

Your Task:

Modify `calc_avg.c` to take command line arguments that match the following example:

```
./a.out nValues fileName
```

Where `nValues` is the number of data points and `fileName` is the file that will be read. You should check that the user has input the correct number of command line arguments. If not, your program should print an error message. Continue to modify the program to read the data from the specified file.

Getting credit for this Lab

To earn credit for this lab, edit `calc_avg.c` to take command line arguments. Submit `calc_avg.c` to *Canvas* under the assignment *Lab2*.