Instructions for using PlotData

1. Executing the program

There are two files: `PlotData.cpp` and `plot`. The latter is a linux shell script written to compile and run `PlotData.cpp` in root. Command line arguments must be supplied to run the program (this is by far the most efficient way to do it). Here is the syntax (order must be correct):

```
$ ./plot <projects> <legends> <title> <time shifts> <time c
ut>
```

Brief summary of arguments (detailed summary below):

- <projects> = project_numChannels or proj1chX,…,projNchY
- <legend> = leg1,…,legN with '_' for spaces. Options: "d", "default"
- <title> = any string with _ for spaces. Options: "", "d", "default"
- <time shifts> = t0 or t0,…tN
- <time cut> = tF

The only required argument is the project(s).

Examples:

```
$ ./plot test
$ ./plot test_3
$ ./plot test 1,2,d, default test_test
$ ./plot test ,,, d 5 10
$ ./plot test1,test3
$ ./plot test1,test2, , test_comparison
$ ./plot test1,test2 d,d title 1,2 8
```

Detailed summary of arguments:

- <projects>: There are two different modes which are parsed differently: mode 0 and mode 1. The mode you are in is automatically detected by the program.
    - mode 0: 1 project name, number of thermometer channels to be used written like <project>_numChannels. This is 4 by default.
    - mode 1: N projects separated by commas (i.e. proj1,proj2,...). In the latter case, the channel number to be used from the project is supplied at the end of each project name. The number of projects is not specified, it is determined from the number of projects listed.
    - In either case, the program recognizes if a file attempting to be accessed does not exist. If there is ever a segfault it is probably related to file-handling.
- <legend>: In either mode, the names of the legends for each channel are specified in order, separated by commas, as with the projects. If no legend is supplied, a simple autolegend will be used. Spaces cannot be used, but underscores will be parsed as

spaces in the plot. “” or “default” will use the default legend text.

- <title>: title of plot. If not entered or supplied “default” or “n”, a default title is created. Spaces cannot be used in the command line, so underscores are parsed as spaces.

- <time shifts>
    - mode 0: shift all times back by t0, and don’t plot any negative times.
    - mode 1: shift the first project specified back by t0, and then shift all other plots back by t1,…,tN (so t0 is an absolute shift of plot 0, the rest are shifts relative to plot 0).
- <time cut>: cut off all points with time above tF (after time shift is applied)

2. About the script

This is not necessary to know unless changing the shell script. Without the script `plot`, the program must be run in root as it uses the root graphics library to plot the data. This is normally done by typing:

```
$ root PlotData.cpp
```

If command line arguments were to be supplied directly, the syntax would be:

```
$ root 'PlotData.cpp(<args>)'
```

The quotes are required (double quotes also work). This is annoying, so there is a linux shell script “plot” written to execute the program with arguments; they are parsed in PlotData.cpp. This concatenates all the arguments into a single string with all args separated by a delimiter: ‘!’.

3. How the program works

Everything important to making the end plot (e.g. legend, title, project(s)) is stored in a `struct plot_args`. All the parsing happens in `plot_args parse_args(string arg_string)`, after which the cases of two modes are handled separately. The rest is routine gui customization and careful bookkeeping.