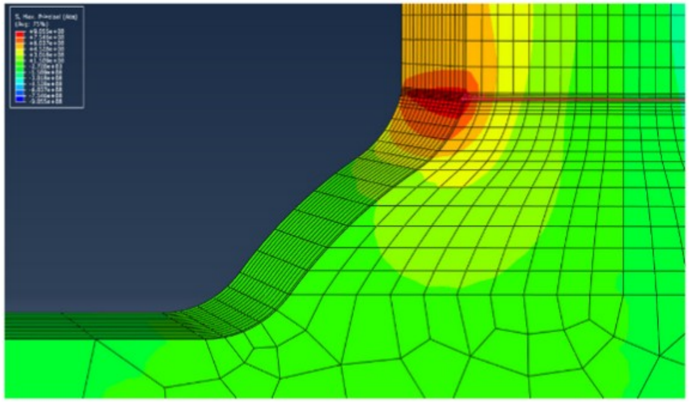
Computer Stuff

Thus far in this course we have only used computers for display, a bit of digitization and some graphing. In the up-coming calculation sections things are going to get much more compute intensive. All the programs and scripts that I use have been developed in Linux. They could be ported to MS Windows, but that will require considerable more effort on your part.

My suggestion to you is to find, borrow or buy a older PC (perhaps you have one stored in your hallway closet? :) and load up a version (32 bit OS is fine) of one of the Linux distributions. This will be your learning machine. We will use it to solve single fatigue hot-spot problems.

This FEA output is for a fairly simple geometry and the hot-spot is obvious.



FEA picture from Fat.Des.Com. of SAE
"Total Life Project"

A car body, however, may have 3million elements.
There may be 60+ load input points on the body.
A proving ground load recording runs for ~2hrs and each of the 60 channels is recorded at 500 samples/sec.
The loads must be superimposed for each element in some way to find all hot-spots.

It is not something done with a spread sheet analysis. It requires hours on a supercomputer.
Most supercomputers run Linux.
The software you learn to use here will work on the large solvers too.

Probably the best way to install a Linux O/S is to have help from someone who has done it before. YouTube also has many instructional videos.
At the moment (2017) the most popular and simple o/s is Linux Mint.

The first decision to make is whether or not you want a "dual boot" system. i.e.: share the computer with both MSWindows and Linux. Sharing is a bit more complicated but seems to work reasonably well. If your install is on a desktop you might consider adding an extra hard drive or SSD; 250Gb is sufficient. The install DVD will probably recognize the new disk and you can avoid extra partitions on your main disk. Also you can then always pull out the 2nd disk if you have problems.
If Linux is the only o/s on the computer the above doesn't matter.

Generally the process is

0. Make a backup of your existing important files.
1. Download the Linux o/s install "iso" file.
2. Burn the iso onto a new DVD
3. In computer BIOS set to boot your DVD drive or USB DVD drive first.
4. Boot the iso DVD and follow the instructions.

Part of the install process will be to set a "root" password, (root is the administrator) and also to specify your user name and password. Avoid running as root for your regular work, as a mistake can cause serious problems. You will need to activate the root user however to install extra software.

Some extra software that you will need: gcc, gfortran, gnuplot, htmldoc, engauge, xv, convert, ….

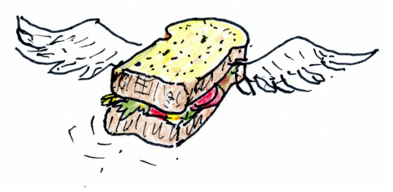
Assuming you now have access to a Linux machine.

1. Open a terminal emulator. The name will depend on your user interface.
   If the interface is "KDE" then open System → Konsole
   If the interface is "Gnome" then open "gnome-terminal"
   (You can adjust the font, background/foreground colors, etc in Konsole
   by right click in Konsole window and "Edit Current Profile".
   In a gnome-terminal use the Edit button in the top bar to change stuff.)

Note: there are many other text file editors/windows available on Linux; many with mouse movement control of where to edit text, much like in MS Word. -but mouse movement editing is too slow if you are going to write software. We are going to use "vi" (also called gvim or vim).

If you are developing software with mouse movement and competing against another developer who uses a keyboard only interface like vi, the vi person will eat your lunch… and probably your next raise.

2. In a web browser find a tutorial on vim or gvim and do it.
   (Probably in this point of your career you know how to "10 finger type" without
   looking at the keyboard? If not you should try some of the typing tutorials
   also available on the web. It will probably take about a day to learn. Keep thinking
   about typing speed and your missing lunch.)



3. In your Konsole or Gnome-term. you can practice using vi simply by
   typing vi filename

Lets try a simple program compile

1. In your terminal (Konsole, etc) change folder to your home folder with:

```
cd
```

then cd (change directory or folder) into your "bin" folder:

```
cd  bin
```

If bin does not exist then create it with `mkdir bin`

Commands (executable files) that you create are often stored in a user's bin folder.

2. With your web-browser download the Fortran program file delete1arg.f :
https://fde.uwaterloo.ca/FatigueClass/Notes/delete1arg.f.txt
into your bin folder.
Change the file's name using mv (move) In terminal window use:

```
ls                                              (see if its there)
mv   delete1arg.f.txt   delete1arg.f
ls                                                   (check)
```

The web browsers need a " .txt " suffix to recognize text files, but the gfortran compiler expects a " .f " suffix, hence the name change

Now lets look at the top of this file using head:

```
>    head    delete1arg.f
```

```
C delete1arg.f  Strip off the first field of each line in a table

C Linux Compile:   gfortran  -g -w -fbounds-check delete1arg.f  -o delete1arg
C Old Linux Compile:  g77 -g delete1arg.f  -o delete1arg
C Sun Compile:  f77 -g -Bstatic delete1arg.f  -o delete1arg
C usage:   delete1arg  <input  >output
C
C-----------------------------------------------------------------------
C  Copyright (C) 2000 SAE Fatigue Design and Evaluation Committee
C  This program is free software; you can redistribute it and/or
```

If you wanted to see 20 lines you could use: head   -20   delete1arg.f
“ tail “ is a similar command to view the bottom of a file.   Try it too.

```
cat delete1arg.f                          (types the whole file )
cp  delete1arg.f    newProg.f            (copies the file to a new file - which you could edit using   vi )
```

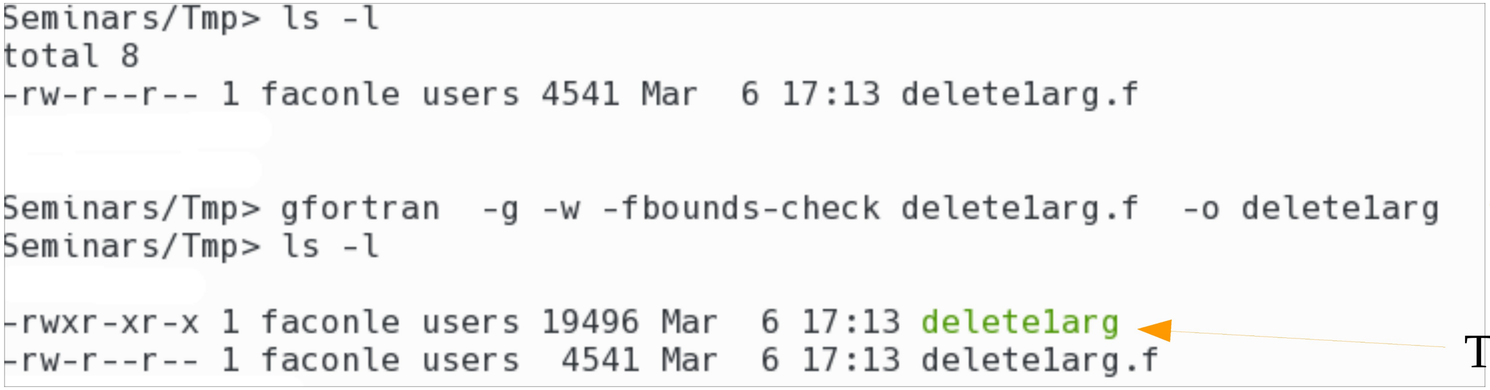
After displaying the top of the file using head , highlight the compile statement using the Left Mouse button

```
> head delete1arg.f
C delete1arg.f  Strip off the first field of each line in a table

C Linux Compile:  gfortran  -g -w -fbounds-check delete1arg.f  -o
delete1arg
C Old Linux Compile:  g77 -g delete1arg.f  -o delete1arg
C Sun Compile:  f77 -g -Bstatic delete1arg.f  -o delete1arg
C usage:   delete1arg  <input  >output
C
C-------------------------------------------------------------------
C  Copyright (C) 2000 SAE Fatigue Design and Evaluation Committee
C  This program is free software; you can redistribute it and/or
```

Then drop it as a command into the terminal window with the Middle mouse button.

(In my case I am working in folder Seminars/Tmp/)



```
Seminars/Tmp> ls -l
total 8
-rw-r--r-- 1 faconle users 4541 Mar  6 17:13 delete1arg.f


Seminars/Tmp> gfortran  -g -w -fbounds-check delete1arg.f  -o delete1arg
Seminars/Tmp> ls -l

-rwxr-xr-x 1 faconle users 19496 Mar  6 17:13 delete1arg
-rw-r--r-- 1 faconle users  4541 Mar  6 17:13 delete1arg.f
```

This file is the executable

Why the need for a program like delete1arg ?

Many of our programs will be writing out more than one type of data. There might be some strain-stress data and some strain-life data. The programs can place a tag at the front of each data line:

outdata

```
#ssdata   0.0   0.0
#ssdata   0.01  350.
#ssdata   -0.01 -350.
#ssdata    0.009  340
#strainLife  0.01 2010.
#strainLife  0.008  5430.
#strainLife  0.005  105200.
```

This is a "pipe" (on keyboard above the "\" char.)

After all the data is written to file outdata one can then process the file to extract only the stress-strain lines, or only the strain life data, using grep and delete1arg to then pipe the data to some different commands. e.g.:

```
grep   \#ssdata    outdata  |   delete1arg  |   cat
grep   \#strainLife   outdata | delete1arg  |  cat
```

(In this case cat just prints the result on screen)

Try this: Create a file that contains the above data: cut and paste or type it into a file named outdata Then run the two grep commands in your terminal window. Then try this:

```
grep \#ssdata  outdata  |   delete1arg |   gnuplot -persist -e 'plot "-" using 1:2 with lp'
```

Having only one output file saves a lot of file clutter in analysis folders.