Trial 2 Hide and Seek, Medium

This is slightly harder than the first question.

This is based on data collected on a particular kind of hack. The cyber security expert in question has written some processing software. IT works for the trial version of the file (HideSmall) but takes for ever on the mail file Hide_and_see.

Its OK to look at the file - it is in CSV format. Here are the headers.

Index	Header name
0	index
1	ts
2	uid
3	id.orig_h
4	id.orig_p
5	id.resp_h
6	id.resp_p
7	proto
8	service
9	duration
10	orig_bytes
11	resp_bytes
12	conn_state
13	local_orig
14	local_resp
15	missed_bytes
16	history
17	orig_pkts
18	orig_ip_bytes
19	resp_pkts
20	resp_ip_bytes
21	tunnel_parents label detailed-label
22	Tunnel
23	Traffic_Labeled
24	Malware

Dont expect to run Hide_and_seek on Non optimised code.

The whole file will take a very long time (several hours) on the original code. My improvements got it down to a minute. Change filename to largeName (Mac or PC) and you should get a proper test.

```
if platform.system() == 'Windows':
    largeName = 'Hide_and_seek_PC.csv.gz'
    smallname = 'HideSmall.csv.mac.gz'
else:
    smallname = 'HideSmall.csv.mac.gz'
    largeName = 'Hide_and_seek.csv.gz'

#smallname = 'HideSmall.csv.gz'
#largeName = 'Hide_and_seek.csv.gz'

filename = smallname
```

When you have your faster version working

```
filename = largeName # change to largeName for heavy test.
```

Please note that you should not decompress the files

HideSmall.csv.mac.gz or
Hide_and_seek.cv.gz or
Hide_and_seek_PC

It is OK to decompress them to look at - nothing is a secret but Uncompressed the file is 271Mb which is big even if it is about the side of data you get in industry.

There is special code which will decompress the file on the fly just before you process it. This is why you should not profile the file reading code.

Problems with byte ordering

This was an issues that came up in testing. This let to the files were not being read in correctly causing errors. again these are the kinds of issues you get in real world data sets and you should start to get used to handling. At the beginning of a file there was a test for which operating system you're running on. This should handle the data sets for you. if not, you're going to have to fix this issue yourself. Given that it's not algorithmic feel free to talk to your tutors in the workshops about the problem.

The solution is to have two different input files one for the PC and one for Unix (Mac). The small size hide and seek small seems to work across both platforms without changes.

How to know when you have made it fast enough.

The real world answer is when you can sit before a huge file and get it processed before you.

I would like to make software to help you (I might release this after the assessment is out) my experience has show that this can cause confusion - and this is bad enough.

When I run Medium_student.py using HideSmall.csv.mac.gz on my machine I get

```
 50551 function calls in 0.041 seconds
```

When I run the optimised version I get

```
 50373 function calls in 0.010 seconds
```

So a speed increase of 0.041 / 0.010 which gives 4.1 or 410% but this is on the small data set. If you are getting about this then you should be able to run on bit data set.

When I run my optimised version on Hide_and_seek.csv.mac.gz I get

```
 21757210 function calls in 79.353 seconds
```

I don't want to wait for the slow version

Look on blackboard for the league table

You can do better than this (even with out a faster machine this was on my laptop) **but you don't need to**.

To help you I will put up a league table. This will have speeds for no improvement , Par (that is as good as my score) and beyond par which I am interested to see.

HideSmall.csv.mac.gz

No improvement	Pro result	High score
1	4.1	