ECE 498 HW1

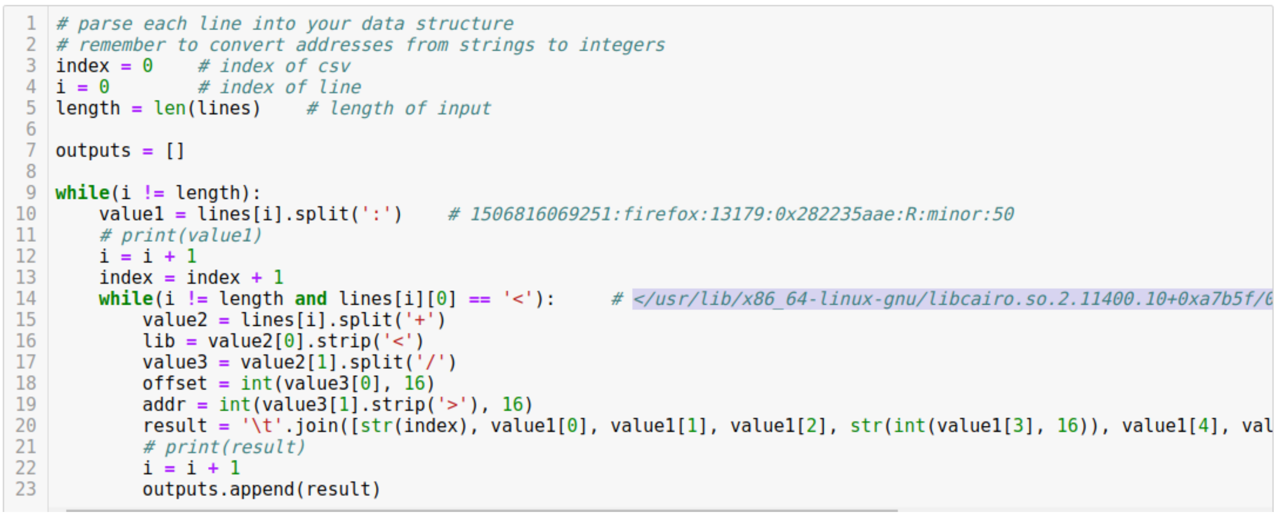
Name: Shuyue Lai
NetID: shuyuel2
Status: Registered

A. Show the data structure you used to parse in the raw log file in terms of python dictionaries, lists, sets, etc. (1 slide)

I use list as the output data structure. The first character of entry if different as below:

- 1506816069251:firefox:13179:0x282235aae:R:minor:50
- </usr/lib/x86_64-linux-gnu/libcairo.so.2.11400.10+0xa7b5f/0xff3f00>

Thus, make it as a while condition to split every entry.
Then, combine values from the first line and lib/addr/offset together as a string.
Last, add this string into the list.



```python
# parse each line into your data structure
# remember to convert addresses from strings to integers
index = 0     # index of csv
i = 0         # index of line
length = len(lines)     # length of input

outputs = []

while(i != length):
    value1 = lines[i].split(':')     # 1506816069251:firefox:13179:0x282235aae:R:minor:50
    # print(value1)
    i = i + 1
    index = index + 1
    while(i != length and lines[i][0] == '<'):      # </usr/lib/x86_64-linux-gnu/libcairo.so.2.11400.10+0xa7b5f/6
        value2 = lines[i].split('+')
        lib = value2[0].strip('<')
        value3 = value2[1].split('/')
        offset = int(value3[0], 16)
        addr = int(value3[1].strip('>'), 16)
        result = '\t'.join([str(index), value1[0], value1[1], value1[2], str(int(value1[3], 16)), value1[4], val
        # print(result)
        i = i + 1
        outputs.append(result)
```

result = '\t'.join([str(index), value1[0], value1[1], value1[2], str(int(value1[3], 16)), value1[4], value1[5], value1[6], lib, str(addr), str(offset)])

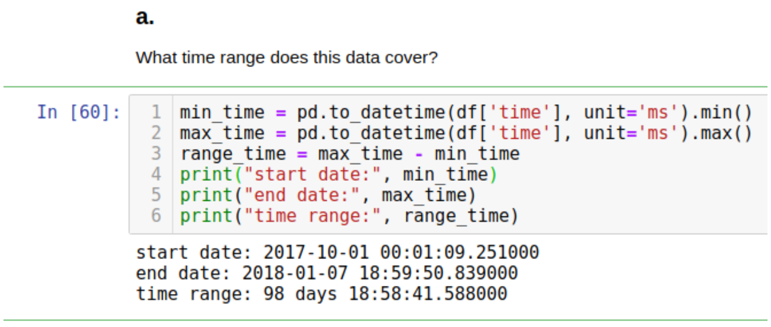
B. Data Analysis

a. What time range does this data cover?

The start date is 2017-10-01 00:01:09.251000

The end date is 2018-01-07 18:59:50.839000

Then range is 98 days 18:58:41.588000



a.

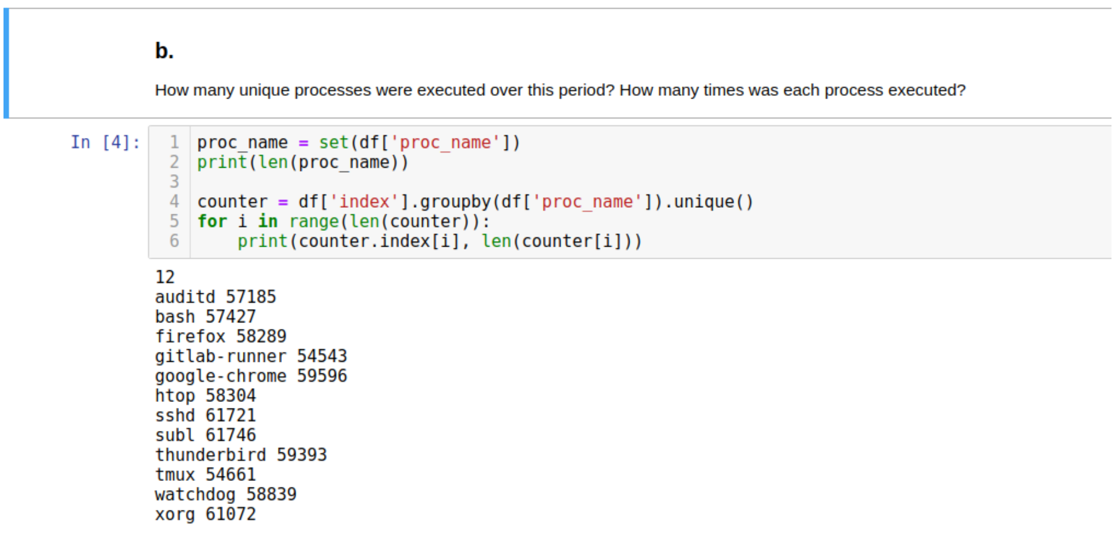
What time range does this data cover?

```
In [60]:
min_time = pd.to_datetime(df['time'], unit='ms').min()
max_time = pd.to_datetime(df['time'], unit='ms').max()
range_time = max_time - min_time
print("start date:", min_time)
print("end date:", max_time)
print("time range:", range_time)
```

```
start date: 2017-10-01 00:01:09.251000
end date: 2018-01-07 18:59:50.839000
time range: 98 days 18:58:41.588000
```

b. How many unique processes were executed over this period? How many times was each process executed?

- 12 unique processes were executed
- auditd 57185
- bash 57427
- firefox 58289
- gitlab-runner 54543
- google-chrome 59596
- htop 58304
- sshd 61721
- subl 61746
- thunderbird 59393
- tmux 54661
- watchdog 58839
- xorg 61072



b.

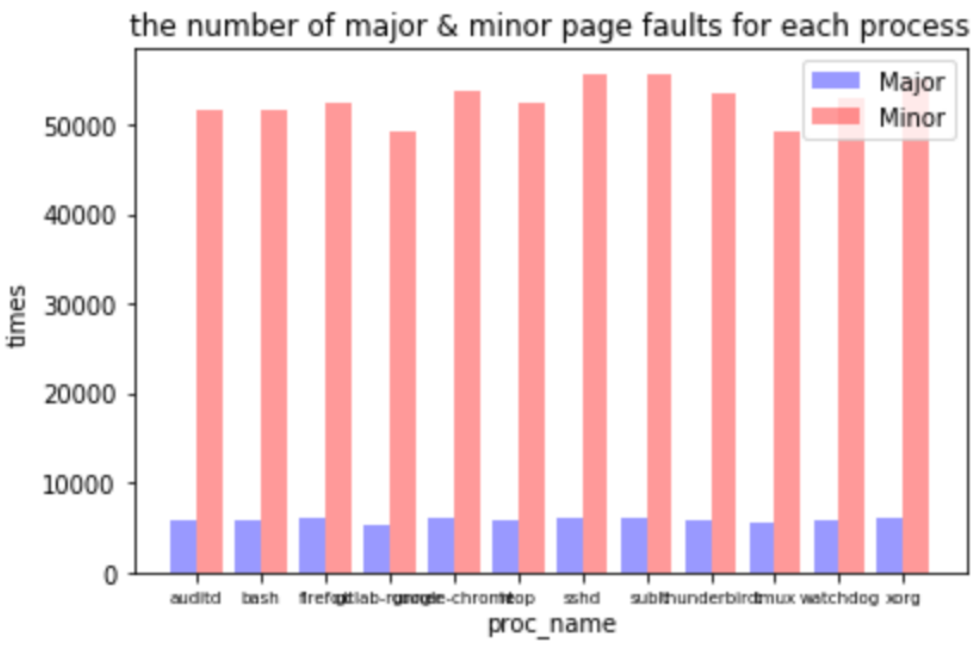
How many unique processes were executed over this period? How many times was each process executed?

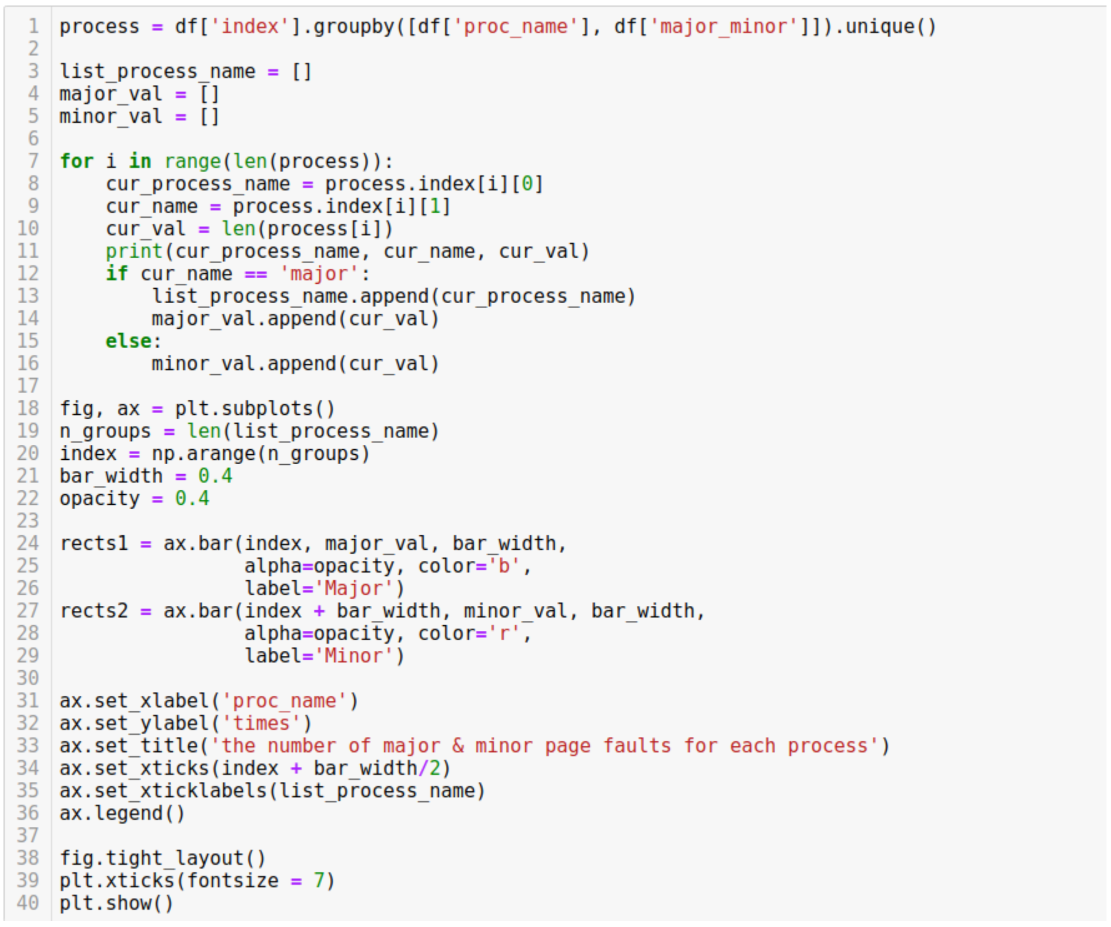
In [4]:
```
proc_name = set(df['proc_name'])
print(len(proc_name))

counter = df['index'].groupby(df['proc_name']).unique()
for i in range(len(counter)):
    print(counter.index[i], len(counter[i]))
```

```
12
auditd 57185
bash 57427
firefox 58289
gitlab-runner 54543
google-chrome 59596
htop 58304
sshd 61721
subl 61746
thunderbird 59393
tmux 54661
watchdog 58839
xorg 61072
```

c. Compare the number of major & minor page faults for each process (averaged over all runs). Plot a bar chart with two categories - major & minor, to demonstrate your results.

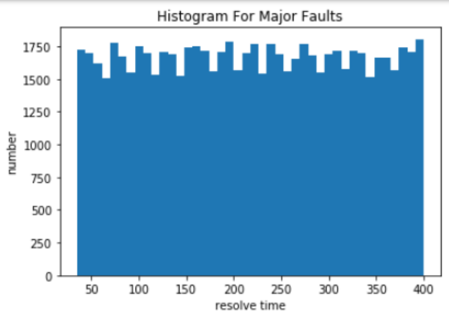
- According to the plot, the number of minor page faults are nearly 10 times to major page faults.

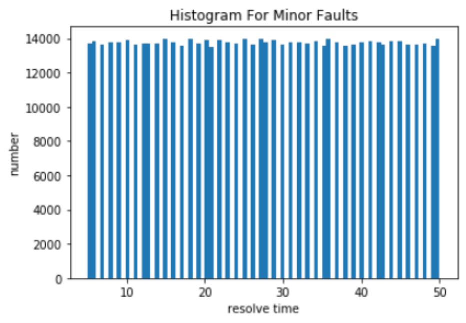




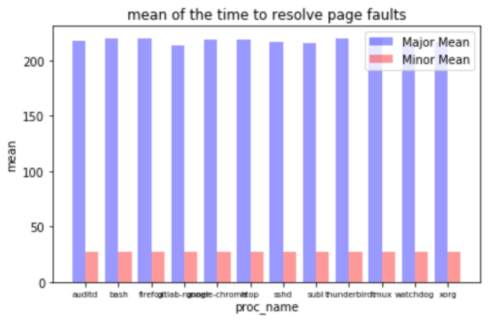
```
process = df['index'].groupby([df['proc_name'], df['major_minor']]).unique()

list_process_name = []
major_val = []
minor_val = []

for i in range(len(process)):
    cur_process_name = process.index[i][0]
    cur_name = process.index[i][1]
    cur_val = len(process[i])
    print(cur_process_name, cur_name, cur_val)
    if cur_name == 'major':
        list_process_name.append(cur_process_name)
        major_val.append(cur_val)
    else:
        minor_val.append(cur_val)

fig, ax = plt.subplots()
n_groups = len(list_process_name)
index = np.arange(n_groups)
bar_width = 0.4
opacity = 0.4

rects1 = ax.bar(index, major_val, bar_width,
                alpha=opacity, color='b',
                label='Major')
rects2 = ax.bar(index + bar_width, minor_val, bar_width,
                alpha=opacity, color='r',
                label='Minor')

ax.set_xlabel('proc_name')
ax.set_ylabel('times')
ax.set_title('the number of major & minor page faults for each process')
ax.set_xticks(index + bar_width/2)
ax.set_xticklabels(list_process_name)
ax.legend()

fig.tight_layout()
plt.xticks(fontsize = 7)
plt.show()
```

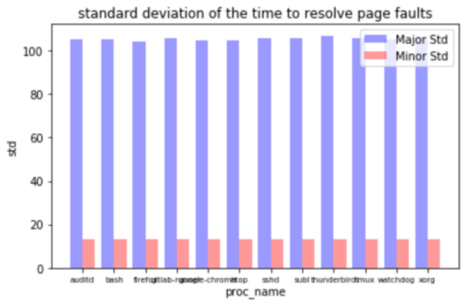
d. Plot the histogram for the time to resolve page faults. Label the axes. For each process, report the mean and standard deviation of the time to resolve page faults. Plotting and calculations should be done for major and minor page faults separately





```
df_index_unique = df.groupby(df['index']).head(1)
major_hist = list(df_index_unique[df_index_unique['major_minor'] == 'major']['resolve_time'])
minor_hist = list(df_index_unique[df_index_unique['major_minor'] == 'minor']['resolve_time'])

```





```
auditd major mean: 217.74495702508332 std: 105.35906622785735
auditd minor mean: 27.52037526221739 std: 13.286257258669712
bash major mean: 217.9333099947488 std: 105.21799066066299
bash minor mean: 27.441137796341415 std: 13.27484236956508
firefox major mean: 220.67785010849607 std: 104.52043936727114
firefox minor mean: 27.571360281463917 std: 13.276757678780305
gitlab-runner major mean: 213.84048384048384 std: 106.11060977256928
gitlab-runner minor mean: 27.37474620319987 std: 13.26370137953019
google-chrome major mean: 218.56416301785112 std: 104.98036266868435
google-chrome minor mean: 27.508144172350814 std: 13.230825904103908
htop major mean: 218.407319552694 std: 104.72779963102408
htop minor mean: 27.42723560169459 std: 13.277470988175734
sshd major mean: 216.40532107078337 std: 105.3689123905007
sshd minor mean: 27.506794650560828 std: 13.29993366952612
subl major mean: 215.4346549192364 std: 105.69754011721432
subl minor mean: 27.445493284427425 std: 13.255274351430309
thunderbird major mean: 220.43844971828582 std: 107.00165023266837
thunderbird minor mean: 27.487018081291094 std: 13.252947868596987
tmux major mean: 218.88298649142024 std: 105.85161912756197
tmux minor mean: 27.447288697314114 std: 13.27651808931501
```

```
watchdog major mean: 217.2808242506812 std: 105.42146631340314
watchdog minor mean: 27.599203277512412 std: 13.269930245395235
xorg major mean: 217.52352747614347 std: 105.73048449828167
xorg minor mean: 27.534931083390916 std: 13.278452551205989
```