OS HW4 report

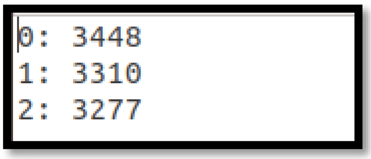
Name: 李品慈

Student ID: 0816039

Q1:

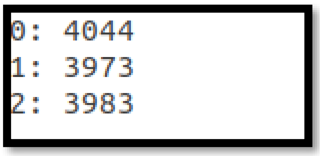
Compare results between hw4_1_1 with/without synchronization.

Without synchronization, the calculated occurrence differ according to the computing process. There may be unexpected race conditions and cause different outputs.



```
0: 3448
1: 3310
2: 3277
```

With synchronization, the calculated occurrence is consistent.

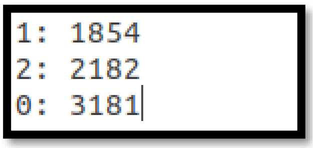


```
0: 4044
1: 3973
2: 3983
```

Q2:

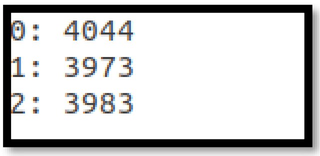
Compare results between hw4_1_2 with/without synchronization.

Without synchronization, the calculated occurrence and the output order may differ.



```
1: 1854
2: 2182
0: 3181
```

With synchronization, the calculated occurrence and output order are consistent.

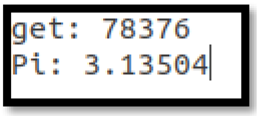


```
0: 4044
1: 3973
2: 3983
```

Q3:

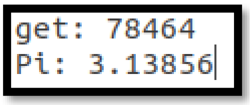
Compare results between hw4_2 with/without synchronization.

Without synchronization, the point we get may be slightly more/less but overall the points are enough to make it ignorable.



```
get: 78376
Pi: 3.13504
```

With synchronization, the point we get maybe slightly more accurate than the above.



```
get: 78464
Pi: 3.13856
```

Q4:

Some problems you meet and how to resolve.
or some Reflections.

I only encountered problems when I'm working on 1_2.
By duplicating the work from 1_1, I didn't really know how to make the first tread wait for the third thread before it output its data. So I tried to replace the mutex with semaphore, but then I met another problem.
The second problem I met is that I'm stuck at adding in the wait and post function, it all became deadlock because the first thread waits for the third thread and vice versa. Then, I tried separating the counting part and output part by assigning two independent semaphores and it finally worked :)