محمد مهدی نظری ۹۹۳۱۰۶۱ – امید خباز قانع ۹۹۳۱۰۱۸

بخش اول)

در این سوال ما ۲ عدد خواننده داریم به همراه یک نویسنده که می خواهند مقداری را بخوانند و بنویسند . مشکل این است که اگر هیچ کنترلی روی خواندن ها و نوشت ها انجام نشود باعث ایجاد حالت مسابقه می شود یعنی در هر بار اجرا خروجی های متفاوتی را مشاهده می کنیم . پس باید از قفل ها استفاده بکنیم تا بتوانیم خواندن و نوشت را کنترل شده انجام بدهیم . در این سوال از دو قفل استفاده می کنیم . نحوه کار هم به این شکل است که وقتی در حال نوشت هستیم به کمک قفل اجازه نمی دهیم تردی بخواند و وقتی در حال خواندن هستیم اجازه نمی دهیم ترد نویسنده مقدار را تغییر دهد . به کمک این روش می توانیم عملیات خواندن و نوشت را کنترل شده انجام بدهیم تا به مشکلی برنخوریم.

کد این بخش در فایل موجود است .

بخش دوم)

در این سوال هم از قفل ها (۵ قفل) استفاده کرده ایم

```
pthread_mutex_t chopstick[5];
```

در این سوال امکان بن بست وجود دارد چرا که فرض کنید هر یک از فیلسوف ها چوب غذای سمت راست خود را بردارند . در این وضعیت همه فیلسوف ها یکی از قفل های خود را در اختیار دارند (همه چوب غذای سمت راست را برداشته اند) و خب به وضوح در این حالت بن بست صورت گرفته است . راه حلی که از آن استفاده شده است به این صورت است که خب اگر ما زمان ها را طوری قرار دهیم که وقتی یک فیلسوف غذاخوردنش تمام شد بعد از آن فیلسوف بعدی بخواهد چوب را بردارد خب در این حالت قطعا به دو فیلسوف اجازه خوردن غذا داده می شود (۴ چوب استفاده بشود) . هدف از این راه این است که اطمینان داشته باشیم زمان خارج شدن چوب از دست یک فیلسوف کمتر از زمان برداشته شدن توسط فیلسوف بعدی باشد و تنها راهی که من به ذهنم رسید این است که زمان غذا خوردن کمتر باشد برای همین در متد run که قرار است هر

یک از فیلسوف ها اجرا کنند یک ۵ sleep قرار داده ام چون این جوری اطمینان داریم که چوب ها زوتر از استفاده آزاد می شوند .توجه کنید که اگر هر ترد را که درست کردیم در main همان جا همه موارد لازم با آن را مشخص کنیم به چنین خروجی بر می خوریم که نشان دهنده اشکال کار است پس باید در سه حلقه مختلف کار های ترد ها init , join , createرا انجام بدهیم.

کد این بخش در فایل موجود است .

خروجی هردو بخش :

```
mmnazari@MMNazari1380: ~/Desktop/OSLab06
mmnazari@MMNazari1380:~$ cd Desktop/OSLab06
mmnazari@MMNazari1380:~/Desktop/OSLab06$ gcc -pthread -o Part01 Part01.c
mmnazari@MMNazari1380:~/Desktop/OSLab06$ ./Part01
PID(W): 3003    count: 1
PID(R): 3004    count: 1
PID(R): 3005    count: 1
PID(W): 3003    count: 2
PID(R): 3004    count: 2
PID(R): 3005    count: 2
mmnazari@MMNazari1380:~/Desktop/OSLab06$ gcc -pthread -o Part02 Part02.c
mmnazari@MMNazari1380:~/Desktop/OSLab06$ ./Part02
Philosopher 5 is thinking.
Philosopher 5 is eating with : chopstick[4] & chopstick[0].
Philosopher 4 is thinking.
Philosopher 3 is thinking.
Philosopher 1 is thinking.
Philosopher 2 is thinking.
Philosopher 5 finished eating.
Philosopher 4 is eating with : chopstick[3] & chopstick[4].
Philosopher 4 finished eating.
Philosopher 3 is eating with : chopstick[2] & chopstick[3].
Philosopher 3 finished eating.
Philosopher 2 is eating with : chopstick[1] & chopstick[2].
Philosopher 1 is eating with : chopstick[0] & chopstick[1].
Philosopher 2 finished eating.
Philosopher 1 finished eating.
```