درس : سیستم های عامل

استاد : طاهری جوان

تمرین10

نام : دانیال کرد مدانلو شماره دانشجویی : 9631813

توضیح روند کار :

پروژه دوم:

سوال 1.1)

نحوه کارکرد قفل کلید در روش پیاده سازی اینجانب به این به صورت می باشد که در هر proccess ای که تمایل دریافت قفل را دارد وارد یک صف می شود و در کنار آن 2 متغیر به صورت گلوبال وجود دارد به نام های turn و ticket و مقادیر اولیه آن ها 0 است و بعد از هر ورود پراسس به صف تنها مقدار ticket اضافه می شود (توسط fetch_and_add که یک تابع در سطح اسمبلی برای اتمیک بودن روند کار می باشد) و هر گاه که یک پراسس خواستار دریافت کلید بود و این 2 مقادیر با هم برابر بودند می تواند به کار خود ادامه دهد(این مقدار قبل از fetch_and_add چک می شود) و گرنه state از RUNNING به WAITING تغییر می کند (که توسط خودم به enum state اضافه شده) و سپس برای اینکه این مقدار ثبت شود و پراسس تا وقتی که نوبتش نشده به کارش ادامه ندهد cpu را از او می گیریم (توسط تابع yield ) در نهایت پس از اتمام کار پراسس تابع release مخصوص کلید قفل صدا زده می شود که در آن یک مقدار به متغیر turn اضافه می شود و متناسب با شماره آن پراسس مربوطه در صف از حالت WAITING بیرون آورده می شود .

## سوال 1.2)

الگوریتم به کار رفته برای این مساله بسیار شبیه آنچه در اسلاید ها موجود است می باشد , به این صورت عمل می کند که یک قفل از جنس کلید داریم و چند متغیر کمکی در کنار آن از جمله تعداد reader هایی که قفل را دارند , هنگامی که یک writer می آید قفل را در هر شرایطی می گیرد و حال اگر کسی قفل را نگرفته بود از قبل می تواند به کارش ادامه دهد و

گرنه صبر می کند تا پراسس های دیگر قفل را آزاد کنند, حال اگر یک پراسس از جنس reader بیاید و صف را خالی مشاهده کند قفل را می گیرد ولی تفائت آن با writer ای می باشد که تنها پراسس اول از رشته پراسس های reader قفل را می گیرد با این روند پراسس های reader می توانند موازی بخوانند و پراسس های writer منتظر می مانند و اگر writer در حال خواندن بود و چند پراسس Reader آمدند همه آن ها قفل را می گیرند و بعد از تمام شدن کار writer تمامی ان ها را باهم آزاد می کند به جای تک تک شان.