بسمه تعالی

پروژه اول
سیستم‌های عامل

- پروژه را به زبان c, c++ و در سیستم‌عامل لینوکس (ترجیحاً Ubuntu برای کسانی که آشنایی چندانی با سیستم‌عامل لینوکس ندارند) انجام دهید
- پروژه به صورت گروهی تعریف شده است.
- **زمان آپلود آن جمعه، ۲۱ آبان ماه** می‌باشد.
- از کپی کردن برنامه‌ها به شدت بپرهیزید.
- در صورت داشتن سوال، می‌توانید از یکی از افراد زیر سوالات خود را بپرسید:
  - m.mahdiani@aut.ac.ir
  - d.alihosseiny@gmail.com
  - amirahmadhabibi@gmail.com
  - mohammadjv94@gmail.com

بخش اول - تمرین فرآیندها و ارتباطات بین فرآیندی

۱. سورس کد سیستم‌عامل آموزشی XV6 را از آدرس زیر دریافت کنید:

https://github.com/mit-pdos/xv6-public

PCB این سیستم‌عامل را یافته و به سوالات زیر پاسخ دهید:

a. Struct مربوط به PCB از چه فیلدهایی تشکیل شده است؟

b. هر یک از متغیرهای زیر در PCB به چه منظوری ذخیره می‌شود؟

i. Sz

ii. State

iii. Context

iv. Ofile

v. Killed

۲. یکی از دو صورت تمرین زیر را به اختیار انتخاب و پیاده‌سازی نمایید. در هر دو صورت زیر، هدف مرتب‌سازی یک آرایه با سایز ثابت ۱۰,۰۰۰ با استفاده از fork است. در این دو تمرین، آرایه به شکل Shared Memory تعریف می‌شود.

a. مرتب‌سازی ادغامی (Merge sort) را با استفاده از fork پیاده سازی نمایید؛ به این معنی که در هر مرحله، پس از تقسیم کردن بخش مورد نظر از آرایه به دو قسمت، مرتب سازی هر قسمت به صورت موازی و با استفاده از fork انجام میگیرد. پس از اتمام کار فرزندان (برای مرتب کردن دو تکه‌ی آرایه) ادغام دو تکه توسط پدر انجام میگیرد.

b. Quick sort را با استفاده از fork پیاده سازی نمایید؛ به این معنی که در هر مرحله، پس از مشخص کردن محل pivot، مرتب سازی دو تکه باقی‌مانده با استفاده از fork و پردازش موازی صورت گیرد.

برای پروژه انتخابی خود، اگر بخش مورد نظر از آرایه، تعداد M یا کمتر عضو داشت، دیگر fork نکنید و به شکل عادی حل مسئله را ادامه دهید.

- برنامه‌تان را به ازای Mهای مختلف اجرا کنید. آیا زمان اجرای برنامه برای Mهای کوچک افزایش می‌یابد؟ علت را توضیح دهید.

می‌تواند از لینک زیر جهت آشنایی با تکه کدهای مربوط به Shared Memory استفاده نمایید:

http://users.cs.cf.ac.uk/Dave.Marshall/C/node27.html

بخش دوم -کلاغ های گرسنه

یک خانواده کلاغ، یک مادر دارد و n جوجه کلاغ و m ظرف غذا که در ابتدا خالی ست. به صورتی که 0<m<=n . هر جوجه کلاغ باید از یک ظرف غذا بخورد و هر ظرف غذا میتواند در هر زمان تنها در اختیار یک جوجه کلاغ باشد. بنابراین بیشتر از m جوجه کلاغ نمیتوانند به صورت همزمان غذا بخورند. کلاغ مادر جداگانه غذا میخورد و نیازی به ظرف غذا ندارد.

در ابتدا همه ظروف خالی هستند و کلاغ مادر خواب است.

هر جوجه کلاغ مدتی بازی می کند و سپس غذا میخورد. قبل از این که هر جوجه کلاغ شروع به غذا خوردن کند باید صبر کند تا یکی از ظروف غذا پر شود. بعد از خوردن غذا، ظرف موردنظر خالی می شود و نیاز به پرکردن مجدد توسط مادر پیدا می کند.

کلاغ مادر خیلی خسته است و نیاز به استراحت دارد. بنابراین، میخوابد تا یکی از جوجه کلاغ ها او را بیدار کند. وقتی این اتفاق بیافتد، کلاغ مادر به دنبال غذا می رود و همه ظزوف را پر می کند. سپس دوباره به خواب می رود. وقتی جوجه کلاغی بخواهد غذا بخورد و همه ظروف غذا را خالی یابد، باید مادر را بیدار کند. همانطور که اشاره شد، زمانی طول میکشد تا کلاغ مادر به دنبال غذا برود و بازگردد و همه ظروف را پر کند. وقتی که همه ظروف پر شد، بیش از m جوجه کلاغ منتظر نمیتوانند غذا بخورند. از آنجایی که کلاغ مادر خیلی خسته است، نمیخواهد زود بیدار شود. پس اصرار دارد که حتماً فقط یکی از جوجه کلاغ هایی که ظروف را خالی یافت می تواند او را بیدار کند (ممکن است ۲ یا چند جوجه کلاغ با ظروف خالی مواجه شوند ولی فقط یکی از آن ها حق دارد مادر را بیدار کند و بقیه باید منتظر پر شدن ظروف غذا شوند).

دقت داشته باشید که هیچ جوجه کلاغی در حالتی که بعضی در حال خوردن هستند و چند ظرف هم خالی ست، مادر را بیدار نمی‌کند و تا خالی شدن همه ظروف صبر می‌کند.

بعد از تهیه t امین غذا، کلاغ مادر بازنشسته می شود و جوجه ها را رها می کند تا زندگی مستقلی تشکیل دهند.

در این سیستم هر کلاغ با یک ترد شبیه سازی می شود و جوجه کلاغ ها روال زیر را در پیش میگیرند:

```
while (1)
{
     Delay();            // play for a while
     ready_to_eat(...);  // get hungry
     Delay();            // eat for a while
     finish_eating(...); // finish eating
                         // do some other thing
}
```

کلاغ مادر روال زیر را اجرا می کند:

```
while (not time to retire)
{
     goto_sleep(...);    // take a nap
     Delay();            // prepare food
     food_ready(...);    // make food available
     Delay();            // do something else
}
```

توابعی که شما پیاده سازی میکنید:

- `ready_to_eat()`
اگر همه ی ظروف غذا خالی باشند، این تابع، فراخوانگر خود را که یک جوجه کلاغ است، قفل می کند. و فقط یکی از جوجه کلاغ هایی که همه ظروف را خالی می یابد باید کلاغ مادر را از خواب بیدار کند. این تابع وقتی فراخوانگر خود را از قفل خارج میکند که ظرفی غذایی پر در اختیار او باشد.

- `finish_eating()`
هر جوجه کلاغ که غذا خوردنش پایان یافت باید این تابع را صدا کند.

- `goto_sleep()`
هر زمان که کلاغ مادر میخواهد بخوابد باید این تابع را صدا بزند.

- `food_ready()`
زمانی که کلاغ مادر همه `m` ظرف را پر از غذا کرد صدا زده میشود.

برای استفاده از ابزارهای همگام‌سازی از کتابخانه‌های **PThread** و **semaphore** استفاده کنید.

پیاده سازی شما باید موقعیت های مهم زیر را به درستی مدیریت کند:

- در هر زمان بیش از `m` جوجه کلاغ در حال خوردن نیستند.
- هر جوجه کلاغ باید زمانی که میخواهد غذا بخورد و ظرفی در اختیارش نیست یا همه ظروف خالی اند، صبر کند.
- زمانی که یک ظرف غیر خالی موجود باشد، یک جوجه کلاغ گرسنه مشغول غذا خوردن می شود.
- هیچ جوجه کلاغ گرسنه ای از ظرف خالی غذا نخورد.
- هر ظرف غذا در هر لحظه میتواند تنها در اختیار یک جوجه کلاغ باشد.
- فقط یکی از جوجه کلاغ ها می تواند مادر را از خواب بیدار کند.
- کلاغ مادر تا زمانی که یکی از جوجه کلاغ ها او را بیدار کند کاری نمیکند.
- وقتی که مادر در حال جمع آوری غذاست تا وقتی که برگردد و دوباره در حالت استراحت قرار گیرد، هیچ جوجه کلاغی او را بیدار نمی کند.
- قبل از این که همه‌ی `m` ظرف پر شوند جوجه کلاغی شروع به خوردن نمیکند.
- بعد از این که ظروف پر شدند، کلاغ مادر باید به جوجه ها اجازه غذا خوردن بدهد و سپس استراحت کند.
- اتمام برنامه باید بعد از زمانی باشد که کلاغ مادر `t`امین بار ظروف را پرکرد و همه ظروف خالی شدند.

شما تنها مجاز هستید برای همگام سازی ها از `semaphore` و `lock  mutex` استفاده کنید.

ورودی و خروجی

- ورودی برنامه شما به این صورت خواهد بود که m (تعداد ظروف غذا) ، n (تعداد جوجه کلاغ ها) و t (تعداد غذا دادن ها) باید از کاربر گرفته شوند و اگر هرکدام از این ورودی ها ۰ داده شوند باید مقدار پیشفرض ۱۰ در آن قرار گیرد. فرض کنید که مقادیر ورودی همیشه بین ۰ تا ۲۰ خواهند بود.

- ساختار خروجی برنامه شما باید مشابه نمونه‌ی زیر باشد.

(توجه داشته باشید که به دلیل پویایی عملکرد رشته ها در هر بار اجرا نتیجه متفاوتی حاصل می شود فلذا منظور از مشابهت با خروجی زیر، ترتیب کار های آمده در آن نیست و فقط اعلام هر عملیات با جملات زیر باشد و ساختار تو رفتگی ها برای هر جوجه کلاغ طبق شماره ی آن با تعداد فاصله مشخص شود)

```
MAIN: There are 10 baby crows, 5 feeding pots, and 8 feedings.
MAIN: Game starts!!!!!
          ......
   Baby crow 3 started.
          ......
Mother crow started.
          ......
      Baby crow 6 started.
          ......
   Baby crow 3 is ready to eat.
          ......
Mother crow takes a nap.
          ......
     Baby crow 5 is ready to eat.
          ......
  Baby crow 2 sees all feeding pots are empty and wakes up the mother.
        Baby crow 8 is ready to eat.
Mother crow is awoke by baby crow 2 and starts preparing food.
 Baby crow 1 is ready to eat.
          ......
Mother crow says "Feeding (1)"
          ......
     Baby crow 5 is eating using feeding pot 3.
          ......
Mother crow takes a nap.
          ......
   Baby crow 3 is eating using feeding pot 1.
          ......
     Baby crow 5 finishes eating.
          ......
          ......
Mother crow says "Feeding (8)"
 Baby crow 1 is ready to eat.
          ......
Mother crow retires after serving 8 feedings. Game ends!!!
```