تکلیف پیاده‌سازی ۱

در یک فایل لیست و نحوه اجرای دستورات مشخص شده است.

```
3     2
```

در سطر اول فایل تعداد فرآیندهای سطح ۱ و سپس تعداد فرآیندهای سطح ۲ مشخص می‌شود. بر اساس مثال فوق برای اجرای کارها فرآیند اصلی (همان برنامه‌ای که در ابتدا اجرا شده) ابتدا ۳ فرآیند ایجاد می‌کند و در مرحله بعد هر کدام از فرآیندهای سطح ۱ برای خود ۲ فرآیند فرزند ایجاد میکنند. اما فرآیندهای سطح بر اساس دستورات ایجاد می‌شوند. بدین شرح که فرآیند اصلی به تعداد فرآیندهای سطح ۲ دستور به هر کدام از فرآندهای سطح ۱ ارسال می‌کند (در مثال بالا ۲ تا به هر کدام). لیست این دستورات با pipe به فرآیند سطح ۱ ارسال میشود. حالا فرآیند سطح ۱ برای هر دستور یک fork زده و با exec دستورات را اجرا میکند. نتیجه اجرای دستورات از طریق پایپ به فرآیند سطح ۱ و سپس با پایپ دیگر به فرآیند اصلی گزارش میشود. کار چاپ و گرفتن ورودی در command line فقط به عهده فرآیند اصلی است.

```
command1  2
command2  arg1 arg2 10
```

در سطرهای بعدی دستورات بهمراه ارگومانها مشخص شده است و ستون آخر در سطر حداکثر مدت زمان اجرای دستور را بر حسب میلی ثانیه بیان میکند. دستورات می‌توانند دستورات واقعی و یا غیر واقعی باشند.

(دستور میتواند در حد یک sleep به مدت زمان تعیین شده باشد و اصلا لازم نیست یک دستور واقعی مثلا shell اجرا شود)

فرآیند اصلی نتایج را بر حسب شماره سطر دستور در خروجی shell نشان میدهد.

۱- این سیستم را با کمک fork , exec, wait و pipe پیاده سازی کنید

۲- مدت زمان کل برای اجرای این دستورات را اندازه‌گیری کرده و با مجموع اندازه زمان اجرای دستورات که خودتان نوشته‌اید مقایسه کنید (مثلا بر اساس دو سطر فوق اجرای پشت سر هم دستورات باید در حدود ۱۲ میلی ثانیه باشد حالا با اجرای چند فرآیندی چه قدر طول کشیده است)

۳- یک مجموعه دستورات را با پیکره‌بندیهای مختلف اجرا کنید و محاسبه کنید مدت زمان اجرای دستورات چقدر نسبت به حالت پشت سر هم چقدر (مثل بخش ۲) بوده است. (مثلا با پیکره‌بندیهای [1,1] که اجرای سریال است و بعد اجرای ... ,[1,3] ,[1,2] و پس از آن اجرای .. ,[2,2] ,[2,1] و ...) در انتها یک نمودار را ترسیم کنید که مشخص کند بهترین زمان اجرا با چه تنظیماتی بوده است)

بخش اضافی ۱

اجرای برخی دستورات با یک احتمال مشخصی (که شما در کد دستور پیاده‌سازی شده می‌نویسید) با خطا مواجه شده و فرآیندهای مرتبط با خطا پایان می‌پذیرند (مثلا exit(1)). در این حالت فرآیندهای والد در هر سطح در صورت مشاهده خطا فرآیندهای جدیدی ساخته و دستورات خطا دار را مجددا اجرا کنند.

بخش اضافی ۲

ارتباط فرآیندها را در تمامی سطوح بجای پایپ با shared memory پیاده‌سازی کنید. تا جایی که می‌توانید از قفل (lock) استفاده نکنید.

نکات تحویل تمرین

۱- پیاده‌سازی باید با زبان C و در سیستم عامل لینوکس انجام گیرد