دانشگاه علم و صنعت ایران
دانشکده کامپیوتر

سیستم های عامل

تغییر Scheduler در Xv6

قسمت اول پروژه:

یک فراخوان جدید به سیستم عامل xv6 اضافه کنید تا برای پیاده سازی scheduler خودتان از آن استفاده کنید.

در مرحله اول باید proc structure کنونی را داخل کد های xv6 گسترش داده و ۴ فیلد جدید به آن اضافه کنید.

stime, etime, iotime, rtime به ترتیب برای:

start time: زمانی که process جدید ساخته میشود

end time: زمانی که پراسس terminate میشود.

run time: مدت زمانی که پراسس در حال اجرا بود است شما باید بعد از هر کلاکی که برای پراسس زده میشود آن را آپدیت کنید. (total respectively of process)

I/O time: مدت زمانی (تعداد کلاکی) که پراسس معطل I/O شده است.

هر کدام از این فیلد ها را در فایل های مختلفی از پروژه باید مقدار دهی کنید.

در مرحله دوم برای extract کردن اطلاعات مورد نظر از کرنل، در جهت پیاده سازی scheduler نیاز داریم که سیستم کال wait موجود را گسترش دهیم:

```
int waitx(int *wtime, int *rtime)
```

دو آرگومان ورودی دارد که پوینتری به integer اند،

rtime تعداد کل کلاک هایی است که پراسس در آن در حال اجرا بوده

wtime تعداد کل کلاک هایی است که پراسس در آن در حال انتظار و waiting بوده است.

قسمت دوم پروژه:

scheduler کنونی xv6 براساس الگوریتم round robin است، شما باید آن را با Scheduler محبوب priority based scheduling عوض کنید، priority based scheduling ،پراسس با بیشترین اولویت را انتخاب و اجرا می کند. در این حالت هم اگر چند پراسس اولویت یکسانی داشتند ما بین آن ها با round robin زمانبندی می کنیم.

اولویت و **priority** هر پراسس بین ۰ تا ۱۰۰ می باشد، مقدار کمتر نشان دهنده ی اولویت بیشتر است و مقدار **default** اولویت ها را ۶۰ در نظر بگیرید

برای تغییر مقدار **priority** یک **system-call** جدید اضافه کنید که بتوانید با آن اولویت پراسس ها را تغییر دهید.

```
int set_priority(int)
```

این **system-call** باید اولویت قدیمی پراسس را خروجی دهد،

زمانی که اولویت هر پراسس تغییر پیدا میکند باید **Rescheduling** دوباره انجام شود. شما می توانید از **system-call** ای به نام **yield()** استفاده کنید.

در فایل توضیحات یک نمونه و یک مقایسه بین زمانبند جدید **priority based** و **round robin** بیاورید.

قسمت سوم پروژه:

یک **Multi-Level Queue Scheduling** را با سه صف پیاده سازی کنید.

صف اول: که صف با اولویت و **priority** بالاتری از صف ۲و ۳ است.

در این صف باید از الگوریتم **guaranteed scheduling policy** استفاده کنید (هر **policy** زمانبندی عادلانه ای را به انتخاب خودتان را می توانید استفاده کنید)

صف دوم: که این صف برای پراسس هایی با اولویت های **medium** میباشد،

در این صف پراسس ها را بر اساس **policy** های زمانبندی **FIFO** و **Round Robin** زمانبندی کنید.

صف سوم: برای پراسس هایی با کمترین اولویت می باشد.

و باید از **policy round robin scheduling** استفاده کنید.

همچنین دقت کنید پراسس با اولویت بیشتر زودتر از پراسس با اولویت کمتر اجرا میشود.

تحویل پروژه:

برای پروژه ی خود یک گزارش آماده کنید گزارش باید شامل موارد زیر باشد:

اولا : باید در گزارش تمامی فایل ها و قسمت هایی از کد که تغییر دادید را خلاصه توضیح دهید که دقیقا چه فایل هایی را تغییر دادید و چه عملکردی در سیستم دارند.

ثانیا : برای قسمت اول باید با مثال و اسکرین شات درستی اجرای system call جدید را نشان دهید.

ثالثا : برای دو قسمت مربوط به scheduler باید با مثال و اسکرین شات درستی اجرای scheduler جدید را در کدتان را نشان دهید.

تمامی فایل هایی که تغییر داده اید را به همراه گزارشتان به صورت فایل zip و با نام os_proj2_firstname_lastname_stdnum بر روی سایت کوئرا بارگذاری کنید.

موفق باشید.