پاییز ۱۳۹۴

پرهام الوانی

۹۲۳۱۰۵۸

تمرین سری پنجم

سیستم‌های عامل

# سوال ۱

در SJF از آنجایی که همواره پردازه‌ای اجرا میشود که کمترین نیاز به پردازنده را دارد، زمان انتظار برای سایر پردازه‌ها به حداقل می‌رسد.

# سوال ۲

FCFS

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T | P5 | P4 | P3 | P2 | P1 |  |
| 15.8 | 7 | 28 | 14 | 20 | 10 | Turn Around |
| 6.4 | 0 | 14 | 13 | 5 | 0 | Waiting |

RR

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T | P5 | P4 | P3 | P2 | P1 |  |
| 17.2 | 7 | 28 | 4 | 30 | 17 | Turn Around |
| 7.6 | 0 | 13 | 3 | 15 | 7 | Waiting |

SJF

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T | P5 | P4 | P3 | P2 | P1 |  |
| 13.4 | 7 | 28 | 1 | 21 | 10 | Turn Around |
| 3.8 | 0 | 13 | 0 | 6 | 0 | Waiting |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T | P5 | P4 | P3 | P2 | P1 |  |
| 20.4 | 7 | 23 | 4 | 45 | 20 | Turn Around |
| 10.6 | 0 | 7 | 3 | 30 | 10 | Waiting |

# سوال ۳

الگوریتم JSF در صورتی که پروسس‌های ورودی همگی نیاز کمی به پردازنده داشته باشند، آنها را اجرا کرده و پردازه‌ای که نیاز بیشتری به پردازنده دارد دچار قحطی میشود.

# سوال ۴

الگوریتم CFS یا Completely Fair Scheduler در کرنل لینوکس یک الگوریتم زمان بند هست که به پروسس‌ها الویت میدهد و از درخت قرمز-سیاه استفاده کرده و در زمان O(1) عمل میکند.

# سوال ۵

زمانبد باید به پروسس‌هایی که استفاده زیادی از CPU دارند زمان بیشتری را اختصاص دهد این در حالی است که باید به پروسس‌هایی که استفاده کمتری از CPU داشته و بیشتر در انتظار IO هستند زمان کمتری از CPU را اختصاص دهد. اگر زمانبند نتواند این کار را به درستی انجام دهد مقدار زیادی از زمان CPU تلف خواهد شد.

# سوال ۶

#include <stdio.h>

int main**(**int argc**,** char **\***argv**[])**

**{**

int i **=** 0**;**

**while** **(**1**)** **{**

i**++;**

printf**(**"%d\n"**,** i**);**

**}**

**return** 0**;**

**}**

#define \_GNU\_SOURCE

#include <stdio.h>

#include <sched.h>

int main**(**int argc**,** char **\***argv**[])**

**{**

int i**,** ret**;**

cpu\_set\_t set**;**

CPU\_SET**(**0**,** **&**set**);**

ret **=** sched\_setaffinity**(**0**,** **sizeof(**cpu\_set\_t**),** **&**set**);**

**if** **(**ret **==** **-**1**)**

perror**(**"sched\_setaffinity()"**);**

i **=** 0**;**

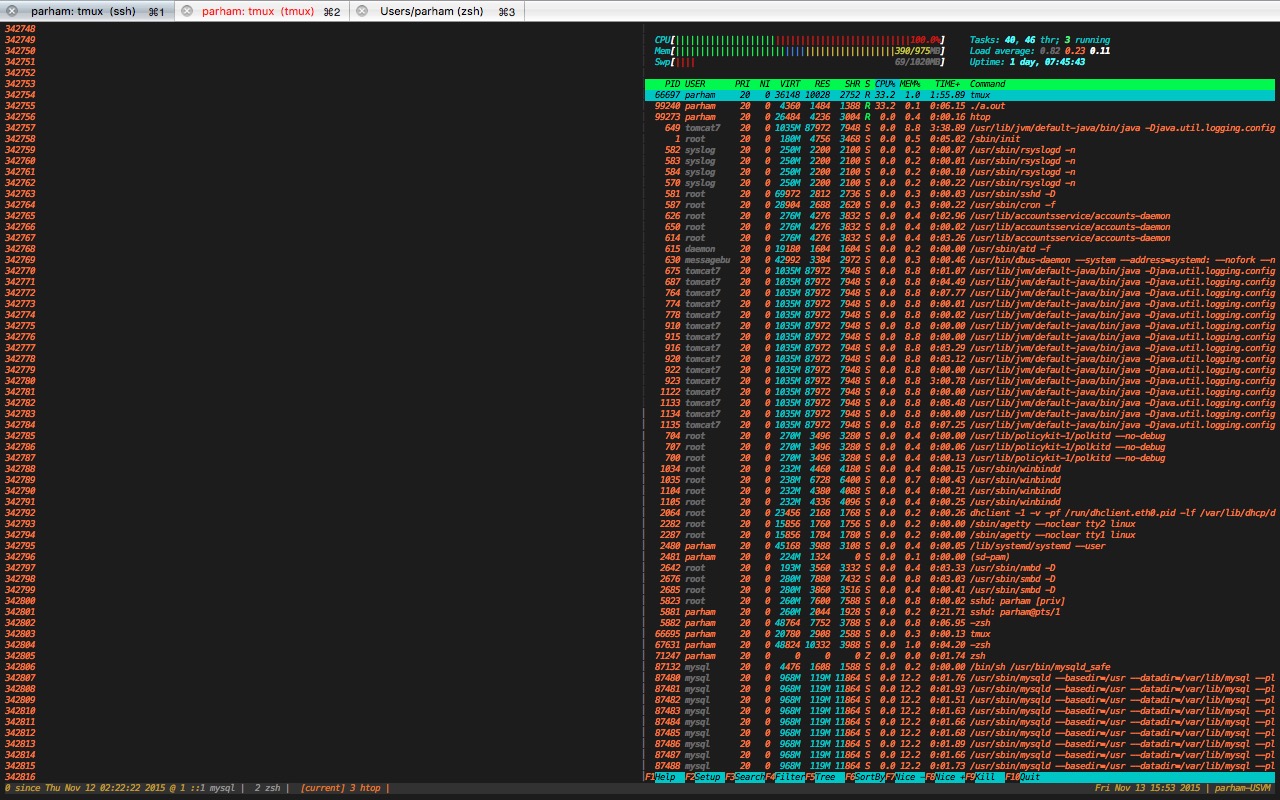
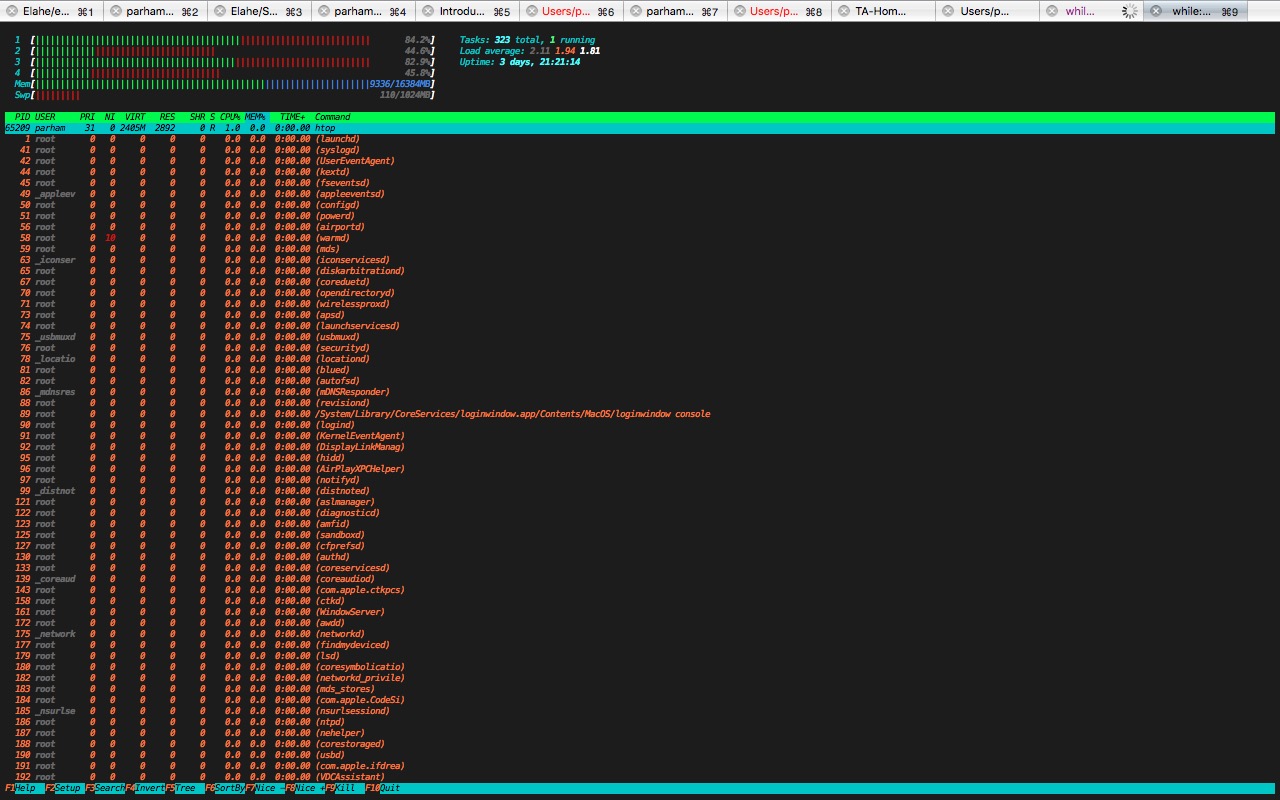
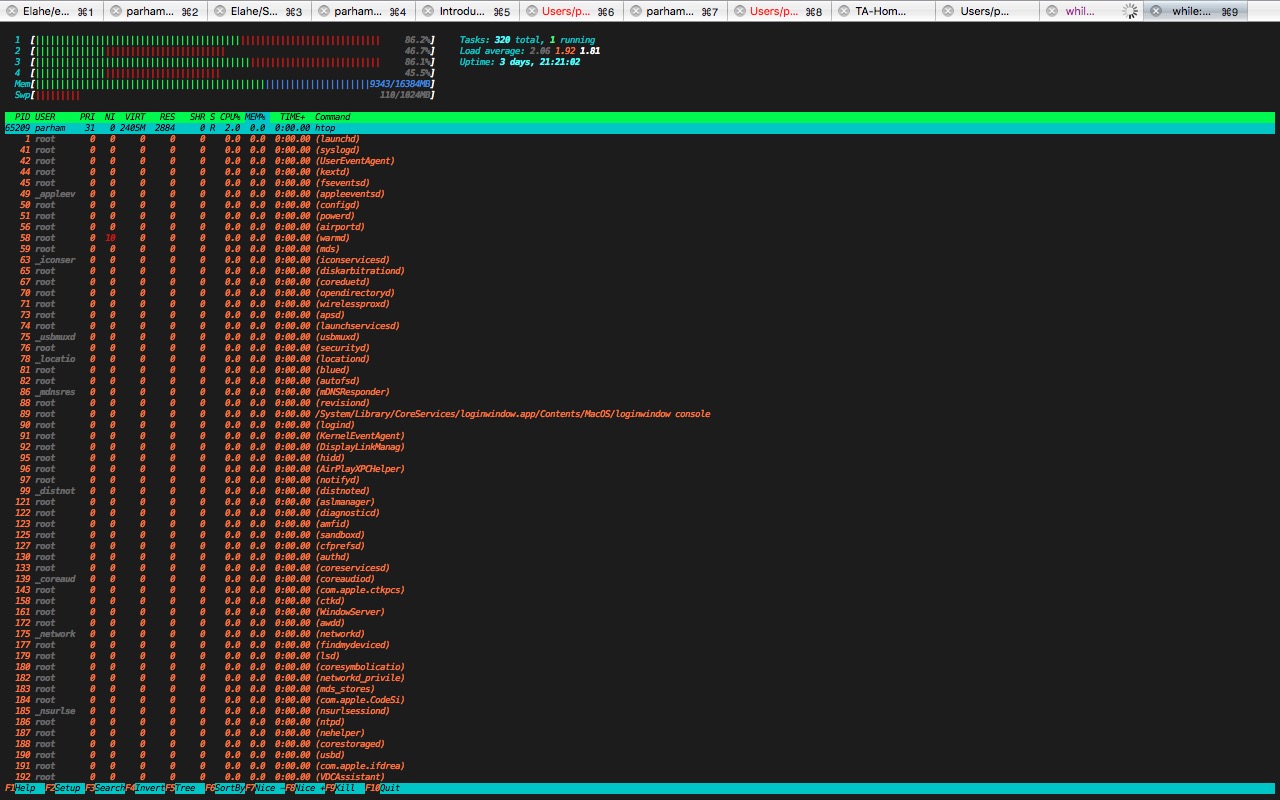
**while** **(**1**)** **{**

i**++;**

printf**(**"%d\n"**,** i**);**

**}**

**}**



# سوال ۷

در SCS مدیریت و زمان بندی threadها بر عهده کرنل میباشد و به این ترتیب هر یک یا چند thread سطح کاربر به تعداد kernel thread مرتبط میشود ولی در مدل PCS مدیریت این threadها بر عهده کاربر و کتابخانه‌ای در سطح کاربر هست.

# سوال ۸

در سیستم‌های بی‌درنگ از آنجایی که نیاز هست در یک بازه زمانی کوتاه به یک event پاسخ داده شود نیاز هست که فاصله زمانی event تا زمان اجرای handler حداقل گردد.