موسسه بابان

انتشارات بابان و انتشارات راهیان ارشد

درس و کنکور ارشد

سیستم عامل

(حل تشریحی سوالات دولتی ۱۳۹۷)

ویژه‌ی داوطلبان کنکور کارشناسی ارشد مهندسی کامپیوتر و IT

براساس کتب مرجع

آبراهام سیلبرشاتز، ویلیام استالینگز و اندور اس تنن‌بام

ارسطو خلیلی‌فر

تست‌های فصل دوم

۹۴- کدام روش بیشترین تسریع را در اجرای یک برنامه ایجاد می‌کند؟

(مهندسی *IT* – دولتی ۹۷)

۱) ۲۰ درصد برنامه ۶۰ درصد تسریع یابد.
۲) ۳۰ درصد برنامه ۵۰ درصد تسریع یابد.
۳) ۵۰ درصد برنامه ۳۰ درصد تسریع یابد.
۴) ۴۰ درصد برنامه ۴ برابر تسریع یابد.

پاسخ‌های فصل دوم

۹۴- گزینه (۴) صحیح است.

مطابق قانون امدال (amdahl) داریم:

$$T_{improved} = [T_{unaffected}] + \left[T_{affected} \times \frac{1}{improvment\ factor} \right]$$

مثال: برنامه‌ای را در نظر بگیرید که زمان اجرای آن 100 ثانیه است، 80 ثانیه‌ی آن مربوط به عملیات ضرب می‌باشد، اگر سرعت عملیات ضرب 60 درصد تسریع پیدا کند، آنگاه زمان اجرا چقدر خواهد بود؟

توجه: به تفاوت کلمه **سرعت عملیات** و **زمان اجرا** دقت کنید، به طور مثال، سرعت عملیات کلی دو برابر شود یعنی زمان اجرای کلی از 100 ثانیه به 50 ثانیه بهبود یابد.

$$T_{improved} = [0.2 \times 100] + \left[0.8 \times 100 \times \frac{1}{1+0.6} \right] = [20] + [50] = 70$$

بنابراین زمان اجرای کلی 30 ثانیه بهبود یافت.

توجه: دقت کنید که 80 ثانیه‌ی عملیات ضرب، سرعت عملیات آن 60 درصد تسریع پیدا کرد و 80 ثانیه به 50 ثانیه بهبود یافت، برای درک بهتر فرض کنید که 80 ثانیه‌ی عملیات ضرب، سرعت عملیات آن 100 درصد تسریع پیدا کند که 80 ثانیه به 40 ثانیه بهبود می‌یابد. یعنی **سرعت** عملیات ضرب دو برابر شده است که به تبع آن **زمان اجرای** عملیات ضرب نصف می‌شود، مطابق رابطه‌ی زیر:

$$T_{improved} = [0.2 \times 100] + \left[0.8 \times 100 \times \frac{1}{1+1} \right] = [20] + [40] = 60$$

توجه: چنانچه در زمان اجرا تمام بخش‌های فرآیند مورد نظر به طور یکنواخت درگیر باشند، می‌توانیم مساله را به شکل ذیر حل کنیم:

توجه: فرض کنیم فرآیندی زمان اجرای آن یک ثانیه طول بکشد.

اگر مطابق گزینه‌ی **اول** عمل نماییم، رابطه‌ی زیر را خواهیم داشت:

$$T_{improved} = [0.8 \times 1] + \left[0.2 \times 1 \times \frac{1}{1+0.6} \right] = [0.8] + [0.125] = 0.925\,s$$

اگر مطابق گزینه‌ی **دوم** عمل نماییم، رابطه‌ی زیر را خواهیم داشت:

$$T_{improved} = [0.7 \times 1] + \left[0.3 \times 1 \times \frac{1}{1+0.5} \right] = [0.7] + [0.2] = 0.9\,s$$

اگر مطابق گزینه‌ی **سوم** عمل نماییم، رابطه‌ی زیر را خواهیم داشت:

$$T_{improved} = [0.5 \times 1] + \left[0.5 \times 1 \times \frac{1}{1+0.3} \right] = [0.5] + [0.384] = 0.884\,s$$

اگر مطابق گزینه‌ی **چهارم** عمل نماییم، رابطه‌ی زیر را خواهیم داشت:

$$T_{improved} = [0.6 \times 1] + \left[0.4 \times 1 \times \frac{1}{4} \right] = [0.6] + [0.1] = 0.7\,s$$

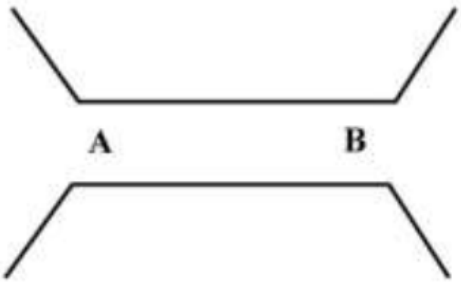
بنابراین واضح است که روش ۴۰ درصد برنامه ۴ برابر تسریع یابد یعنی گزینه‌ی چهارم، **بیشترین تسریع را در زمان اجرای یک برنامه ایجاد می‌کند. بنابراین گزینه‌ی چهارم پاسخ سوال است.**

تست‌های فصل ششم

۹۵- مسئله پل را در نظر بگیرید که هر خودرو برای عبور باید از هر دو سر پل به ترتیب عبور کند. اگر تعداد زیادی خودرو در هر دو طرف این پل برای عبور از آن وجود داشته باشد و در هر لحظه امکان عبور فقط یک خودرو باشد. کدام مورد درست‌تر است؟

(مهندسی IT – دولتی ۹۷)

۱) این مسئله فقط با سمافور قابل حل است.
۲) این مسئله فقط با مانیتور قابل حل است.
۳) این مسئله نه با سمافور و نه با مانیتور قابل حل است.
۴) این مسئله هم با سمافور و هم با مانیتور قابل حل است.



پاسخ‌های فصل چهارم

۹۵- گزینه (۴) صحیح است.

صورت سوال به این شکل است:

مسئله پل را در نظر بگیرید که هر خودرو برای عبور باید از هر دو سر پل به ترتیب عبور کند. اگر تعداد زیادی خودرو در هر دو طرف این پل برای عبور از آن وجود داشته باشد و در هر لحظه امکان عبور فقط یک خودرو باشد. کدام مورد درست‌تر است؟

در هر لحظه امکان عبور فقط یک خودرو یعنی برقراری شرط انحصار متقابل برای عبور از پل، که برقراری شرط انحصار متقابل توسط هم سمافور و هم مانیتور امکان‌پذیر است.

۱) این مسئله فقط با سمافور قابل حل است.

گزینه اول نادرست است، زیرا برقراری شرط انحصار متقابل توسط هم سمافور و هم مانیتور امکان‌پذیر است.

۲) این مسئله فقط با مانیتور قابل حل است.

گزینه دوم نادرست است، زیرا برقراری شرط انحصار متقابل توسط هم سمافور و هم مانیتور امکان‌پذیر است.

۳) این مسئله نه با سمافور و نه با مانیتور قابل حل است.

گزینه سوم نادرست است، زیرا برقراری شرط انحصار متقابل توسط هم سمافور و هم مانیتور امکان‌پذیر است.

۴) این مسئله هم با سمافور و هم با مانیتور قابل حل است.

گزینه چهارم درست است، زیرا برقراری شرط انحصار متقابل توسط هم سمافور و هم مانیتور امکان‌پذیر است.

توجه: سازمان سنجش آموزش کشور، ابتدا در کلید اولیه خود گزینه دوم را به عنوان پاسخ اعلام نمود، اما در کلید نهایی گزینه‌ی چهارم را به عنوان پاسخ نهایی اعلام نمود، که کار درستی بوده است.

تست‌های فصل دوم

۹۶- در یک سیستم بی‌درنگ، سه فرایند به‌صورت زیر به‌طور متناوب تولید می‌شود. در این نمایش، 35^* بدین معنی است که فرایند P_2 پس از ۳۵ واحد زمانی از لحظه ایجاد، دوباره تکرار می‌شود، موعد (Deadline) آن برابر ۱۰۰ (۳۵ + ۶۵) است. در صورتی که از الگوریتم غیرقبضه‌ای «اول زودترین موعد» (Earliest-deadline first) برای زمان‌بندی استفاده شود، اگر فرآیند P_4 در زمان صفر با دوره تناوب ۱۰۰ واحد زمانی ایجاد شود، برای آنکه زمان‌بندی سیستم امکان‌پذیر باشد، حداکثر مقدار Burst Time آن کدام است؟

Process	Arrival	Burst Time	Period/deadline
P_1	@0	20	50
P_2	@15	25	65+35*
P_3	@25	20	90

۱) ۵
۲) ۱۰
۳) ۱۵
۴) مقداری وجود ندارد.

پاسخ‌های فصل دوم

۹۶- گزینه (۴) صحیح است.

زمان‌بندی تمام وقایع متناوب مستقل از نوع زمان‌بندی در یک سیستم بی‌درنگ به شرطی امکان‌پذیر است که رابطه‌ی زیر برقرار باشد:

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 1$$

C_i : زمان اجرای فرآیند

P_i : دوره تناوب

مطابق فرض سوال، فرآیند P_2 پس از 35 واحد زمانی از لحظه‌ی ایجاد، دوباره تکرار می‌شود، که موعد (Deadline) آن برابر100 است. بنابراین فرآیند P_2 یکبار در لحظه‌ی 15 با موعد 100 واحد زمانی ایجاد می‌شود و یکبار هم در لحظه‌ی 35 با موعد 100 واحد زمانی ایجاد می‌شود که این سناریو به طور متناوب تکرار می‌شود. بنابراین مطابق رابطه‌ی فوق داریم:

$$\sum_{i=1}^{m} \frac{C_i}{P_i} = \left\lceil \frac{C_1}{P_1} \right\rceil + \left\lceil \frac{C_{21}}{P_{21}} + \frac{C_{22}}{P_{22}} \right\rceil + \left\lceil \frac{C_3}{P_3} \right\rceil + \left\lceil \frac{C_4}{P_4} \right\rceil =$$

$$\left\lceil \frac{20}{50} \right\rceil + \left\lceil \frac{25}{100} + \frac{25}{100} \right\rceil + \left\lceil \frac{20}{90} \right\rceil + \left\lceil \frac{x}{100} \right\rceil = 0.4 + 0.5 + 0.22 + \left\lceil \frac{x}{100} \right\rceil \leq 1$$

توجه: مطابق فرض سوال، فرآیند P_4 در زمان صفر با دوره تناوب 100 واحد زمانی ایجاد می‌شود. واضح است که حاصل جمع عبارت $0.4+0.5+0.22$ از مقدار یک بزرگتر است یعنی $0.4+0.5+0.22>1$ بنابراین رابطه‌ی زیر برقرار نیست:

$$0.4 + 0.5 + 0.22 + \left\lceil \frac{x}{100} \right\rceil \leq 1$$

بنابراین برای فرآیند P_4 برای آنکه زمان‌بندی سیستم امکان‌پذیر باشد، مقداری برای Burst Time یا زمان اجرای آن وجود ندارد.

توجه: در سیستم‌های بی‌درنگ به دلیل تناوبی بودن اجرای فرآیندها، زمان ورود به سیستم معنا ندارد و در رابطه‌ی فوق لحاظ نمی‌گردد.

تست‌های فصل سوم

۹۷- در سیستم‌های تعبیه شده بی‌درنگ سخت که پاسخ در زمانی مشخص باید تضمین شود، کدام روش نگاشت ریسمان‌های کاربر به ریسمان‌های سیستمی، مناسب است؟

(مهندسی *IT* – دولتی ۹۷)

۱) یک به یک
۲) چند به یک
۳) چند به چند
۴) دوسطحی

پاسخ‌های فصل چهارم

۹۷- گزینه (۴) صحیح است.

به طور کلی مدیریت و زمان‌بندی نخ‌ها به سه روش زیر انجام می‌گردد:

۱- روش سطح کاربر یا مدل چند به یک (many to one)

در این روش فقط زمان‌بند پردازنده و زمان‌بند چند نخی در سطح کاربر وجود دارد و زمان‌بند چند نخی در سطح هسته در این روش مورد استفاده قرار نمی‌گیرد. در واقع هسته سیستم عامل فقط فرآیندها را می‌شناسد و هیچ اطلاعاتی از نخ‌ها ندارد. در واقع اولویت‌بندی نخ‌ها، مدیریت نخ‌ها و زمان‌بند چندنخی در سطح کاربر و توسط یک بسته نرم‌افزاری انجام می‌گردد. بدین معنی که نخ‌ها را برنامه‌نویس مشخص می‌کند و مدیریت آن‌ها را نیز بر عهده می‌گیرد. بنابراین زمان‌بند پردازنده، براساس الگوریتم مشخصی مثلاً نوبت چرخشی پردازنده را در اختیار یکی از فرآیندهای آماده قرار می‌دهد. سپس زمان‌بند چند نخی در سطح کاربر، متناسب با کاربردی که در آن فرآیند به کار گرفته می‌شود، الگوریتم زمان‌بند را انتخاب کرده و تصمیم می‌گیرد که پردازنده در اختیار کدام‌یک از نخ‌های آماده در فرآیند موردنظر قرار گیرد و تا زمانی که پردازنده در تملک فرآیند باشد و یا تا قبل از پایان برش زمانی مربوط به فرآیند، نخ‌های یک فرآیند از پردازنده بهره‌مند می‌شوند و به محض مسدود شدن یک نخ، و یا پایان برش زمانی یک فرآیند، یا اتمام فرآیند، پردازنده به فرآیند بعدی تعلق می‌گیرد.

توجه: در این روش نخ ماهیت منطقی دارد و از دید کاربر فقط وجود دارد، در واقع از نظر سیستم عامل ماهیت فیزیکی ندارد، بنابراین نخ کاربر در این روش همانند یک تابع در فرآیند می‌باشد که از رجیستر و پشته مختص به خود نیز بهره‌مند نمی‌باشد.

توجه: مدل غیرکامپیوتری این روش نیز وجود دارد، مانند حالتی که درآمدهای دولت حاصل از منابع کشور، بین پدران خانواده‌ها تقسیم گردد و این پدران خانواده‌ها باشند که تصمیم بگیرند به هر عضو خانواده چه مقدار نقدینگی تعلق بگیرد. در این روش فقط پدران شماره حساب مختص به خود را دارند. اما اگر یکی از اعضای خانواده خطایی انجام دهد و محکوم گردد، آنگاه تمام اعضای خانواده برای مدتی از خدمات دولت محکوم می‌گردند، زیرا در این مدل، دولت فقط پدران خانواده را می‌شناسد و از اعضای خانواده اطلاعی ندارد. بنابراین حساب پدر خانواده برای مدتی مسدود می‌گردد.

توجه: عمل تعویض متن مابین نخ‌های یک فرآیند کاربر، کاملاً در سطح کاربر و با سربار بسیار ناچیز (در حد فراخوانی نخ بعدی) و بدون تغییر حالت پردازنده به مد هسته پردازنده، توسط زمان‌بند چندنخی در سطح کاربر (برنامه‌های کاربر) انجام می‌گردد. اما عمل تعویض متن مابین فرآیندها (فرآیندهای کاربر یا فرآیندهای سیستم عامل) در سطح هسته سیستم عامل و در مد هسته

پردازنده، توسط زمان‌بند پردازنده برای انتخاب یک فرآیند جدید بر اساس یک الگوریتم خاص انجام می‌گردد.

توجه: به دلیل آنکه نخ‌های یک فرآیند کاربر، تماماً در فضای کاربر مدیریت می‌شوند، اگر نخ موجود در یک فرآیند کاربر، یک فراخوان سیستمی مسدودکننده را اجرا نماید، هسته سیستم عامل نه تنها آن نخ، بلکه کل فرآیند کاربر را که شامل تمام نخ‌های دیگر می‌باشد، مسدود می‌کند، زیرا هسته سیستم عامل خبری از نخ‌های داخل فرآیند کاربر ندارد. در واقع در این روش هسته سیستم عامل نخ‌ها را همانند توابع داخل یک فرآیند می‌بیند، یعنی هسته سیستم عامل، یک فرآیند چند نخی سطح کاربر را، مانند یک فرآیند تک نخی اما دارای چند تابع مختلف می‌بیند!

توجه: این راه‌کار، منجر به عدم امکان هم‌روندی (در سیستم‌های تک‌پردازنده‌ای) نخ‌های داخل یک فرآیند کاربر در حالت انسداد یک نخ داخل یک فرآیند کاربر می‌گردد.

البته اگر نخی داخل یک فرآیند کاربر مسدود نگردد، امکان هم‌روندی میان نخ‌های داخل فرآیند کاربر برقرار است. مانند یک تیم فوتبال ۱۱ نفره که اگر بازیکنی مرتکب خطا گردد، از آن‌جا که داور فقط نام تیم را می‌شناسد و نه تک‌تک بازیکنان تیم را، آن‌گاه کل تیم را جریمه، اخراج و مسدود می‌کند. اما اگر هیچ‌یک از بازیکنان تیم مرتکب خطایی نگردد، واضح است که هم‌روندی برقرار است.

توجه: فرض کنید یک کیک داریم که آن را به چهار قسمت مساوی تقسیم کرده‌ایم، همچنین فرض کنید خوردن هر بخش کیک یک ساعت زمان بخواهد، اگر یک نفر بخواهد تمام این کیک را بخورد، پس ۴ ساعت طول می‌کشد اگر ۴ نفر بخواهند تمام این کیک را بخورند و به هر نفر یک بخش کیک داده شود، آنگاه خوردن تمام کیک به‌طور موازی ۱ ساعت طول خواهد کشید. در روش سطح کاربر، یک فرآیند چند نخی کاربر نمی‌تواند از امتیازات چند پردازنده‌ای بهره ببرد، زیرا در روش سطح کاربر، هسته سیستم عامل در هر لحظه فقط یک پردازنده را در اختیار نخ‌های یک فرآیند کاربر قرار می‌دهد. حتی اگر چند پردازنده موجود باشد. یعنی در این روش خوردن کیک بخش‌بندی شده به صورت چند نفری امکان‌پذیر نیست. بنابراین در این روش امکان پردازش موازی نخ‌های داخل یک فرآیند کاربر وجود ندارد.

توجه: نرم‌افزارهای POSIX P-threads و Mach C-thread به عنوان یک بسته نرم‌افزاری، می‌توانند جهت مدیریت نخ‌ها و زمان‌بند چند نخی در سطح کاربر مورد استفاده قرار گیرند.

توجه: در این روش تخصیص منابع و زمان‌بندی پردازنده، بر روی فرآیندها انجام می‌شود. همچنین زمان‌بندی نخ‌های سطح کاربر، بر عهده زمان‌بند چندنخی سطح کاربر خواهد بود.

توجه: سیستم عامل سولاریس، مدل چند به یک را پیاده سازی می‌کند.

۲- روش سطح هسته یا مدل یک به یک (one to one)

در این روش فقط زمان‌بند پردازنده و زمان‌بند چندنخی در سطح هسته وجود دارد و زمان‌بند

چند نخی در سطح کاربر در این روش مورد استفاده قرار نمی‌گیرد. در واقع هسته سیستم عامل نه تنها فرآیندها را می‌شناسد بلکه از وجود نخ‌های داخل یک فرآیند نیز در صورت وجود نخ آگاه است. در واقع مدیریت نخ‌ها و زمان‌بند چندنخی (نخ‌های فرآیندهای کاربر و نخ‌های فرآیندهای سیستم عامل) در سطح هسته، توسط هسته سیستم عامل انجام می‌گردد و کاربر هیچ دیدی از این کار ندارد. انگار که اجتماع تمام نخ‌های فرآیندهای سیستم عامل و کاربر را در نظر بگیرید، حال بر روی تک نخ‌ها بر اساس یک الگوریتم خاص حرکت کنید. بنابراین زمان‌بند چند نخی در سطح هسته، براساس الگوریتم مشخصی مثلاً نوبت چرخشی پردازنده را در اختیار یکی از نخ‌های آماده قرار می‌دهد. توجه کنید در این روش پردازنده دیگر در تملک فرآیند نیست بلکه در تملک نخ‌ها است. در واقع تا زمانی که پردازنده در تملک یک نخ باشد و تا قبل از مسدود شدن نخ، و یا تمام شدن نخ و یا پایان برش زمانی مربوط به نخ، نخ می‌تواند از پردازنده بهره ببرد و به محض مسدود شدن یک نخ، یا پایان برشی زمان مربوط به نخ یا اتمام نخ، پردازنده می‌تواند به نخ بعدی که ممکن است، نخ بعدی، نخی هم‌خانواده با نخ قبلی در یک فرآیند باشد، یا نخی در یک فرآیند دیگر باشد، تعلق بگیرد. در واقع زمان‌بند چند نخی در سطح هسته سیستم عامل تعیین می‌کند که نخ بعدی که باید شروع به کار کند متعلق به همان فرآیند باشد و یا از یک فرآیند دیگر انتخاب شود.

توجه: برای انجام زمان‌بندی چند نخی، هسته سیستم عامل باید علاوه بر جدول فرآیندها، یک جدول نخ (شبیه جدول فرآیند) داشته باشد که اطلاعات تمامی نخ‌های موجود در سیستم را نگهداری کند. مجدداً تأکید می‌کنیم که در این حالت هر نخ TCB خاص خود را دارد. به عبارت دیگر هر نخ رجیستر و پشته مختص به خود را دارد.

توجه: در این روش نخ ماهیت فیزیکی دارد، و از دید کاربر و سیستم عامل وجود دارد، بنابراین در این روش، هر نخ، رجیستر و پشته مختص به خود را دارد، به عبارت دیگر هر نخ TCB مختص به خود را دارد.

توجه: عملیات مرتبط با نخ‌های سطح کاربر، مانند ایجاد (بارگذاری TCB مختص به نخ) و پایان دادن (ذخیره‌سازی TCB مختص به نخ) بر عهده هسته سیستم عامل است و توسط زمان‌بند چند نخی در سطح هسته سیستم عامل انجام می‌گردد.

توجه: مدل غیرکامپیوتری این روش نیز وجود دارد، مانند حالتی که درآمدهای دولت حاصل از منابع کشور، بدون اعمال هیچ‌گونه اولویت‌بندی بین تک تک اعضاء یک کشور تقسیم گردد (حال این اعضاء شهروند عام باشد یا خاص) در این روش هر یک از اعضای کشور شماره حساب مختص به خود را دارند. اما اگر یکی از افراد کشور خطایی انجام دهد و محکوم گردد، آنگاه فقط همان فرد برای مدتی از خدمات دولت محروم می‌گردد و دولت به بقیه اعضاء خانواده آن فرد همچنان خدمات ارائه می‌دهد. زیرا دولت از مشخصات تک تک اعضاء جامعه آگاه است.

توجه: عمل تعویض متن، مابین نخ‌های یک فرآیند (فرآیندهای کاربر یا فرآیندهای سیستم

عامل) در سطح هسته سیستم عامل، در مد هسته پردازنده توسط زمان‌بند چندنخی در سطح هسته سیستم عامل انجام می‌گردد و عمل تعویض متن مابین فرآیندها (فرآیندهای کاربر یا فرآیندهای سیستم عامل) در سطح هسته سیستم عامل و در مد هسته پردازنده توسط زمان‌بند پردازنده انجام می‌گردد.

توجه: به دلیل آنکه نخ‌های یک فرآیند (فرآیندهای کاربر یا فرآیندهای سیستم عامل)، تماماً در فضای هسته سیستم عامل مدیریت می‌شوند و هسته سیستم عامل از وجود نخ‌های یک فرآیند آگاه است، اگر نخ موجود در یک فرآیند، یک فراخوان سیستمی مسدود کننده را اجرا نماید، هسته سیستم عامل فقط آن نخ مربوطه را مسدود می‌کند و نخ‌های دیگر هم‌خانواده با آن نخ مسدود شده، همچنان می‌توانند از پردازنده بهره ببرند.

توجه: این راه کار، منجر به امکان هم‌روندی (در سیستم‌های تک پردازنده‌ای) و امکان توازی (در سیستم‌های چندپردازنده‌ای) میان فرآیندهای مختلف و یا میان نخ‌های داخل یک فرآیند می شود. البته اگر نخی مسدود نگردد، درجه همروندی و توازی بالاتر هم خواهد رفت، زیرا در اینصورت همه نخ‌ها به طور هم‌روند یا موازی در حال حرکت هستند. مانند یک تیم فوتبال ۱۱ نفره که اگر بازیکنی مرتکب خطا گردد، همروندی یا توازی همچنان برقرار است، چون داور تک تک بازیکنان را می‌شناسد و فقط بازیکن خاطی را مسدود، جریمه و اخراج می‌کند و بقیه بازیکنان تیم به بازی خود ادامه می‌دهند، اما اگر هیچ یک از بازیکنان تیم مرتکب خطا نگردند، واضح است که همروندی و توازی بالاتر هم خواهد بود.

توجه: مثال کیک مطرح شده را مجدداً به یاد آورید، در روش سطح هسته، یک فرآیند چندنخی (فرآیندهای کاربر یا فرآیندهای سیستم عامل) می‌تواند از امتیازات چندپردازنده‌ای بهره ببرد. در روش سطح هسته، هسته سیستم عامل می‌تواند در هر لحظه چندین پردازنده را در اختیار نخ‌های یک فرآیند قرار دهد. یعنی در این روش خوردن کیک بخش‌بندی شده به صورت چند نفری امکان‌پذیر است. بنابراین در این روش امکان پردازش موازی نخ‌های داخل یک فرآیند وجود دارد.

توجه: در این روش، منابع به فرآیندها اختصاص می‌یابد ولی زمان‌بندی پردازنده، بر روی نخ-ها انجام می‌گیرد و زمان‌بندی نخ‌های سطح کاربر و نخ‌های سطح هسته، برعهده زمان‌بند چندنخی سطح هسته می‌باشد.

توجه: سیستم عامل لینوکس، خانواده سیستم عامل ویندوز و سولاریس ۹ مدل یک به یک را پیاده‌سازی می‌کنند.

۳- روش ترکیبی (سطح کاربر و هسته) یا مدل چند به چند (many to many)

این روش از اجتماع دو روش سطح کاربر و هسته ابداع گردیده است. در این روش علاوه بر زمان‌بند پردازنده و زمان‌بند چندنخی در سطح هسته سیستم عامل، زمان‌بند چند نخی نیز در سطح

کاربر برای زمان‌بندی و اولویت‌دهی نخ‌های فرآیند کاربر وجود دارد. در واقع زمان‌بندی نخ‌های فرآیندهای سیستم عامل، توسط هسته سیستم عامل و زمان‌بندی نخ‌های فرآیندهای کاربر، توسط برنامه کاربر انجام می‌گردد، که این امر منجر به اولویت‌بندی نخ‌های فرآیندهای کاربر می‌گردد. در این روش، زمان‌بند چند نخی در سطح کاربر، نخ‌های کاندید خود را از میان نخ‌های متعدد در فرآیندهای مختلف کاربر براساس یک الگوریتم خاص انتخاب و تحویل یک زمان‌بند چند نخی در سطح هسته می‌دهد. در واقع انتخاب نخ‌های کاندید (زمان‌بندی) در فرآیندهای کاربر به خود کاربر واگذار شده است که همانطور که گفتیم منجر به اولویت‌بندی نخ‌های فرآیندهای کاربر می‌گردد. حال اجتماع حاصل از نخ‌های کاندید فرآیندهای کاربر و نخ‌های فرآیندهای سیستم عامل توسط زمان‌بند چندنخی هسته سیستم عامل براساس یک الگوریتم خاص زمانبندی می‌شود.

از توضیحات فوق این مفهوم برداشت می‌شود که هسته سیستم عامل باید این قابلیت را داشته باشد که کاربر بتواند نخ‌های فرآیندهای سطح خود را به هسته سیستم عامل معرفی کند. بنابراین در این روش هسته سیستم عامل نه تنها فرآیندهای کاربر را می‌شناسد، بلکه از وجود نخ‌های کاربر داخل فرآیندهای کاربر نیز در صورت وجود نخ آگاه است.

توجه: برای معرفی نخ‌های کاندید فرآیندهای سطح کاربر به زمانبند چندنخی در سطح هسته سیستم عامل، علاوه بر جایگاه‌های مخصوص نخ‌های فرآیندهای سیستم عامل در زمان‌بند چندنخی در سطح هسته سیستم عامل، تعدادی جایگاه، ویژه نخ‌های سطح کاربر نیز، در زمانبند چندنخی در سطح هسته سیستم عامل در نظر گرفته شده است، به این جایگاه **ویژه که محیط اجرای نخ** نیز نامیده می‌شود، **LWP** گفته می‌شود. در واقع هر نخ کاندید انتخاب شده توسط زمانبند چندنخی در سطح کاربر، پس از معرفی به زمانبند چندنخی در سطح هسته، در یکی از جایگاه‌های ویژه که همان LWP است، جهت زمان‌بندی توسط زمان‌بند چندنخی در سطح هسته، قرار می‌گیرد.

توجه: LWP سرواژه‌ی عبارت Light Weight Process و به معنی فرآیند سبک وزن است.

توجه: هنگامی که یک نخ به زمان‌بند چندنخی در سطح هسته معرفی می‌گردد و در ادامه در یک LWP جهت زمان‌بندی توسط زمان‌بند چندنخی در سطح هسته، قرار می‌گیرد، در طول حیات خود ممکن است، در LWP های متفاوتی بخش‌هایی از اجرای خود را طی کند، مثلاً یک نخ در صورت رسیدن به عملیات ورودی و خروجی، جایگاه خود یعنی LWP را واگذار می‌کند و در اجرای بعدی پس از پایان عملیات ورودی و خروجی ممکن است به یک LWP دیگر منتسب شود. توجه کنید که **LWP محیط اجرای نخ** می‌باشد.

توجه: برای انجام زمان‌بندی چندنخی، هسته سیستم عامل باید علاوه بر جدول فرآیندها، یک جدول نخ (شبیه جدول فرآیند) داشته باشد که اطلاعات تمامی نخ‌های موجود در سیستم را نگهداری کند.

توجه: در این روش نخ ماهیت فیزیکی دارد و از دید کاربر و سیستم عامل وجود دارد،

بنابراین در این روش، هر نخ، رجیستر و پشته مختص به خود را دارد، یـه عبـارت دیگـر هـر نـخ TCB مختص به خود را دارد.

توجه: عملیات مرتبط با نخ‌های سطح کاربر، مانند ایجاد (بارگذاری TCB مخـتص بـه نـخ) و پایان دادن (ذخیره‌سازی TCB مختص به نخ) بر عهده هسته سیستم عامل است و توسط زمـان‌بنـد چندنخی سطح هسته انجام می‌گردد.

توجه: مدل غیرکامپیوتری این روش نیز وجود دارد، مانند حالتی که درآمدهای دولـت حاصـل از منابع کشور، با اعمال نوعی اولویت‌بندی بین برخی از اعضاء یک کشور تقسیم گردد (حال ایـن اعضاء شهروند عام باشند یا خاص) در این روش هریک از اعضـاء اولویـت‌دار و منتخـب کشـور شماره حساب مختص به خود را دارند. اما اگر یکی از اعضاء خانواده خطایی انجام دهد و محکوم گردد، آنگاه فقط همان فرد برای مدتی از خدمات دولت محروم می‌گردد و دولت بـه بقیـه اعضـاء خانواده آن فرد، همچنان خدمات ارائه می‌دهـد، زیـرا دولـت از مشخصـات تـک تـک اعضـاء آن خانواده آگاه است.

توجه: عمل تعویض متن مابین نخ‌های یک فرآیند (فرآینـدهای کـاربر یـا فرآینـدهای سیسـتم عامل) در سطح هسته سیستم عامل، در مد هسته پردازنده توسط زمانبند چندنخی در سـطح هسـته سیستم عامل انجام می‌گردد و عمل تعویض متن مابین فرآیندها (فرآیندهای کـاربر یـا فرآینـدهای سیستم عامل) در سطح هسته سیستم عامل و در مد هسته پردازنده توسط زمانبنـد پردازنـده انجـام می‌گردد.

توجه: به دلیل آنکه نخ‌های کاندید فرآیندهای سطح کاربر، به هسته سیستم عامـل معرفـی مـی گردند و هسته سیستم عامل از وجود نخ‌های کاندید فرآیندهای سطح کاربر آگاه است، اگر یک نخ کاندید موجود در یک فرآیند کاربر، یک فراخـوان سیسـتمی مسـدودکننده را اجـرا نمایـد، هسـته سیستم عامل فقط آن نخ مربوطه را مسدود می‌کند و نخ‌های دیگر هـم‌خـانواده بـا آن نـخ مسـدود شده، همچنان می‌تواند از پردازنده بهره ببرند. این راه‌کار، منجر به امکان هم‌روندی (در سیستم‌های تک‌پردازنده‌ای) و امکان توازی (در سیستم‌های چندپردازنده‌ای) میان فرآیندهای مختلـف کـاربر و یا میان نخ‌های داخل یک فرآیندکاربر می‌شود. البته اگر نخی مسدود نگـردد، درجـه هـم‌رونـدی و توازی بالاتر هم خواهد رفت، زیرا در اینصورت همه نخ‌ها به طور هـم‌رونـد یـا مـوازی در حـال حرکت هستند.

مانند یک تیم فوتبال ۱۱ نفره که اگر بازیکنی مرتکب خطا گردد، هم‌روندی یا تـوازن همچنـان برقرار است، چون داور تک تک بازیکنان را می‌شناسد و فقط بازیکن خاطی را مسـدود، جریمـه و اخراج می‌کند و بقیه بازیکنان تیم به بازی خود ادامه می‌دهند، امـا اگـر هـیچ‌یـک از بازیکنـان تـیم مرتکب خطا نگردد، واضح است هم‌روندی و توازی بالاتر هم خواهد رفت.

توجه: همانند نخ‌های سطح کاربر، همروندی و توازی برای نخ‌های سطح هسته سیسـتم عامـل

نیز، در حالت انسداد یک نخ یا عدم انسداد یک نخ، برقرار است.

توجه: مثال کیک مطرح شده را مجدداً به یاد آورید، در روش ترکیبی، یک فرآیند چندنخی (فرآیندهای کاربر یا فرآیندهای سیستم عامل) می‌تواند از امتیازات چندپردازنده‌ای بهره ببرد. در روش ترکیبی، هسته سیستم عامل می‌تواند در هر لحظه چندین پردازنده را در اختیار نخ‌های یک فرآیند قرار دهد. یعنی در این روش خوردن کیک بخش‌بندی شده به صورت چند نفری امکان پذیر است. بنابراین در این روش امکان پردازش موازی نخ‌های داخل یک فرآیند وجود دارد.

توجه: در این روش نیز، منابع به فرآیندها اختصاص می‌یابد ولی زمان‌بندی پردازنده، بر روی نخ‌ها، انجام می‌گیرد. اما زمان‌بندی نخ‌های سطح کاربر، بر عهده زمان‌بند چندنخی سطح کاربر و زمان‌بندی نخ‌های سطح هسته، بر عهده زمان‌بند چندنخی سطح هسته خواهد بود.

در نرم‌افزارهای بی‌درنگ باید خروجی و پاسخ نهایی در یک زمان مشخص و از پیش تعیین شده حاصل شود. در این نرم‌افزارها، زمان نقشی کلیدی ایفا می‌کند و زمان پاسخ باید به موقع و تضمین شده باشد. نرم‌افزارهای بی‌درنگ معمولاً به عنوان یک دستگاه کنترلی در یک کاربرد خاص (مثلاً صنعتی) به کار گرفته می‌شوند. در این نرم‌افزارها دیر پاسخ دادن به همان بدی پاسخ ندادن است. در این نوع نرم‌افزارها هدف اصلی طراحان، پاسخگویی سریع (در مهلت تعیین شده) به رویدادها و درخواست‌ها می‌باشد و راحتی کاربران و بهره‌وری منابع در درجه‌های بعدی اهمیت، قرار دارند. نتیجه اینکه زمان پاسخ در سیستم‌های بی‌درنگ الزاما باید به موقع و تضمین شده باشد. در طرف مقابل، در سیستم‌های اشتراک زمانی و عمومی، داشتن زمان پاسخ کوتاه مطلوب است ولی الزامی نیست.

به طور کلی سیستم‌های بی‌درنگ به دو نوع زیر طبقه‌بندی می‌شوند:

۱- سیستم بی‌درنگ سخت (Hard Real-Time)

۲- سیستم بی‌درنگ نرم (Soft Real-Time)

در سیستم‌های بی‌درنگ سخت، ضرب‌العجل‌ها یا مهلت زمانی(deadline) باید تحت هر شرایطی رعایت شود، مانند ترمز اتومبیل، باید گرفته شود و زمان بسیار نزدیک است این بی‌درنگ سخت است، دیر پاسخ دادن به همان بدی پاسخ ندادن است، نباید دیر پاسخ دهد یا دستگاه کنترل ضربان قلب انسان، اگر ضربان نبض نبود همه رو باید سریع بیدار کند و نباید دیر پاسخ دهد. در سیستم‌های بی‌درنگ سخت، معمولاً وسایل ذخیره‌سازی ثانویه همچون دیسک به دلیل کندی آن وجود ندارد و به جای آن از حافظه‌های ROM استفاده می‌شود. سیستم عامل‌های **پیشرفته** نیز در این سیستم‌ها وجود ندارد چرا که سیستم عامل کاربر را از سخت‌افزار جدا می‌کند و این جداسازی باعث عدم قطعیت در زمان پاسخگویی می‌شود. به دلیل نیاز به پاسخ‌دهی سریع و تضمین شده سیستم‌های بی‌درنگ از حافظه مجازی استفاده نمی‌کنند. در سیستم‌های بی‌درنگ سخت مهلت زمانی(deadline) باید پشتیبانی شود. در برخی کاربردها (مثل کنترل صنعتی) در کامپیوترها از سیستم عامل استفاده نمی‌شود. از آنجا که در سیستم‌های کنترل صنعتی برنامه می‌بایست در اسرع

وقت در مقابل یک اتفاق، از خود عکس العمل نشان دهد، وجود واسط سیستم عامل باعث کند شدن مراحل می‌گردد. البته در سیستم‌های بی‌درنگ سخت می‌تواند سیستم عامل باشد، اما سیستم عامل باید با شرایط سیستم‌های بی‌درنگ سخت سازگار باشد یعنی سیستم عامل هم بی‌درنگ باشد. سیستم‌عامل بی‌درنگ نوعی سیستم‌عامل است که در آن، زمان پارامتر کلیدی است. سیستم بی‌درنگ به سیستمی گفته می‌شود که درستی اجرای یک عملیات در آن تنها به درست بودن عملیات از نظر منطقی بستگی نداشته باشد بلکه اجرای آن عملیات در یک بازه زمانی مشخص نیز در درستی اجرای عملیات در نظر گرفته شود. در سیستم‌های بی‌درنگ سخت (hard real-time) یا به عبارتی سیستم‌های بی‌درنگ بدون وقفه (immediate real-time) پایان اجرای یک عملیات پس از ضرب‌الاجل بی‌فایده تلقی می‌شود و نوشدارو پس از مرگ سهراب است.

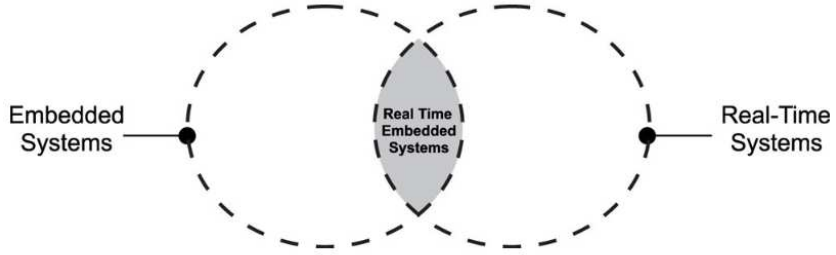
اما در سیستم‌های بی‌درنگ نرم، زیر پاگذاشتن ضرب‌الاجل‌ها با اینکه نامطلوب است اما قابل تحمل است. یعنی می‌توان با چند لحظه تاخیر نیز کنار آمد. مثل رزرو بلیط هواپیما در صورت خالی شدن لحظه‌ای یک صندلی، خوب است اولویت و حق رعایت شود و فورا رزرو انجام شود، ولی اگر هم ضرب‌الاجل‌ها گاها رعایت نشد خیلی هم فاجعه بار نیست و چنین تاخیری قابل تحمل است یا هشدار پیام خالی شدن کاغذ یک دستگاه فتوکپی، خوب است پیام خالی شدن کاغذ فورا اعلام شود، ولی اگر هم ضرب‌الاجل‌ها گاها رعایت نشد خیلی هم فاجعه بار نیست و چنین تاخیری قابل تحمل است. سیستم‌های پخش زنده صدا و تصویر نیز معمولاً سیستم‌های بی‌درنگ نرم هستند که در صورت عدم پاسخگویی سیستم در ضرب‌الاجل با پایین آوردن کیفیت صدا و تصویر وضعیت را مدیریت می‌کنند. نتیجه اینکه در سیستم‌های بی‌درنگ نرم، رعایت مهلت زمانی مطلوب است، ولی اجباری نیست و تضمین هم نمی‌شود. به بیان دیگر سیستم تلاش می‌کند که کار خود را در مهلت زمانی خاص انجام دهد، ولی اجباری هم در این کار نیست. و حتی انجام کار پس از پایان مهلت زمانی، بازهم معنا دارد. پس در سیستم بی‌درنگ سخت احتمال تاخیر زمانی تا پس از مهلت زمانی به طور قطعی و تضمین شده وجود ندارد. اما در سیستم بی‌درنگ نرم احتمال تاخیر زمانی تا پس از مهلت زمانی وجود دارد.

توجه: می‌توان گفت یک سیستم بی‌درنگ سخت تضمین می‌کند که کارها و وظایف بحرانی به موقع انجام شود، اما در یک سیستم بی‌درنگ نرم، یک وظیفه بحرانی نسبت به سایر وظایف اولویت خیلی بالاتری دارد و تا پایان تکمیل شدنش این ارجحیت را حفظ می‌کند. در واقع در سیستم‌های بی‌درنگ سخت پس از پایان مهلت زمانی، ادامه و تکمیل یک کار، دیگر معنی ندارد. از آنجا که سیستم‌های بی‌درنگ نرم مهلت زمانی(deadline) را پشتیبانی نمی‌کنند، استفاده آنها در کنترل صنعتی ریسک آور است. هر چند که سیستم‌های بی درنگ نرم می‌بایست پاسخی سریع داشته باشند ولی مساله پاسخ‌دهی به حادی سیستم‌های بی‌درنگ سخت نمی‌باشد.

انسان در وادی زندگی نیازهای گوناگونی دارد، یکی از نیازهای اساسی انسان، نیاز به امنیت است. اما گاهاً، ممکن است در معرض عوامل محیطی و بیرونی و یا حتی درونی امنیت انسان در

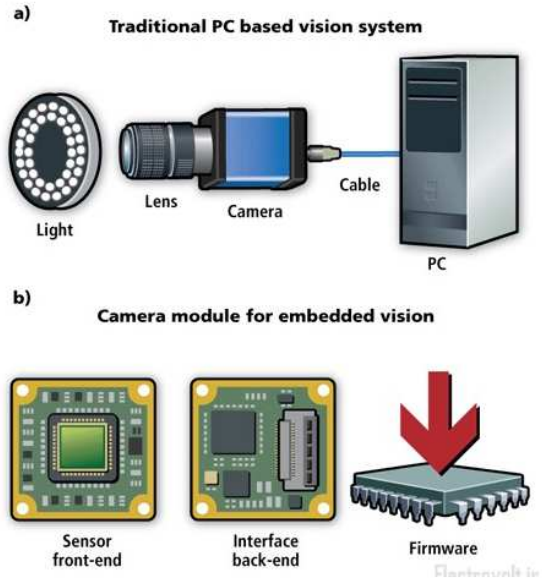
شرایط هشیاری یا ناهشیاری به مخاطره بیفتد. بنابراین نیاز است تـا مکـانیزمی همـواره هوشـیار و همیشه بیدار و با اشراف لحظه به لحظه، مخاطرات پیرامون انسان را رصد و تحت کنترل خود قرار دهد تا در موقع لزوم و به صورت آنی، بی‌درنگ، در لحظه و در زمـان حقیقـی و واقعـی (تـا دیـر نشده) با تهدید مقابله کند، نرم‌افزارهای بی‌درنگ این نگهبان همیشه هوشیار و همیشه بیدار هستند. مانند نرم‌افزارهای ترمز اتومبیل، کنترل ضربان قلب اتاق بیهوشی، کنترل فشار کابین هواپیما و ...

در عصر حاضر دو مفهوم جدید وجود دارد که باعث بوجود آمدن نسل جدیدی از سیستم‌های پرکاربرد شده است. این دو مفهوم یکـی «سیسـتم‌های نهفتـه» یـا Embedded Systems و دیگـری «سیستم‌های بی‌درنگ» یا Real-Time System می‌باشند. در اغلب اوقات این دو سیستم به صورت تلفیقی و تحت عنوان «سیسـتم‌های نهفتـه بی‌درنـگ» یـا Real-Time Embedded Systems مـورد استفاده قرار می‌گیرند. این سیستم‌ها توانایی کنترل دامنه‌ی وسیعی از وسایل مکانیکی و الکترونیکی را دارا می‌باشند. اگر به اطراف خـود نگـاهی بیاندازیـد، انـواع مختلـف آنهـا را مشـاهده می‌کنیـد. تلویزیون، لوازم منزل و آشپزخانه، تلفن‌های همراه هوشمند، سیستم‌های کنترل اتومبیل، سیستم‌های کنترل ترافیک، سیسـتم‌های اتوماسـیون صـنعتی، ربات‌هـا، موشـک‌های نظـامی و ... همـه و همـه مثال‌هایی از این سیستم‌ها می‌باشند. سیستم‌های **نهفته** که به آنها سیستم‌های **تعبیه شده** یا **توکار** نیز گفته می‌شود، سیستم‌های کامپیوتری هستند که شامل اجزای الکترونیکی و یا مکانیکی می‌باشـند و وظیفه‌ی مشاهده (Monitor)، پاسخ دادن (Respond) و کنترل (Control) محیط خارجی سیستم را بـر عهـده دارد. ایـن محـیط خـارجی بوسـیله دسـتگاه‌های ورودی و خروجـی نظیـر سنسـورها (Sensors)، عمل‌کننـده‌ها (Actuators) و ... بـا سیسـتم‌های نهفتـه ارتبـاط دارد. علـت نام‌گـذاری سیستم‌های نهفته (Embedded Systems) این است کـه در گذشـته بـا نگـاه کـردن بـه درون یـک سیستم مثل PC و Laptop نگاه ناظر می‌توانست اجزای فیزیکی داخلی آن نظیـر CPU و RAM را ببیند، اما در سیستم‌های نهفته این اجزا درون IC هستند و روی بورد الکترونیکی بـه نـوعی پنهـان شده است و به صورت مجزا قابل مشاهده نیست.



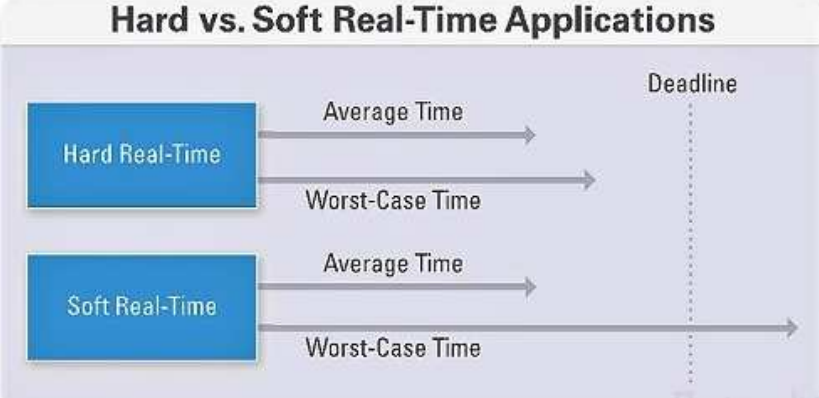
بر خلاف کامپیوترهای همه منظوره (بـه عنـوان مثـال کامپیوترهـای شخصـی) کـه بـرای رفـع نیازهای عمومی طراحی شده‌اند، سیستم‌های نهفته به گونه‌ای طراحی می‌شوند که برای یک کاربرد خاص با **کمترین هزینه، بهتـرین کـارایی** را از خـود نشـان دهنـد. بنـابراین مشخصـه‌ی کلیـدی، سیستم‌های نهفته، طراحی اختصاصی برای انجام یک کار مشـخص اسـت. از آنجاکـه سیسـتم‌های

نهفته برای یک کار مشخص اختصاص یافته‌اند، مهندسین طراح می‌توانند محصول را برای کاهش اندازه و قیمت بهینه کرده و اطمینان‌پذیری و کارایی آنرا بالا ببرند، برخی از سیستم‌های نهفته با بهره‌گیری از مزیت‌های تولید با تیراژ بالا و به تبع مقرون به صرفه‌بودن هزینه‌های تولید، به شکل انبوه تولید شده‌اند. امروزه درون اکثر وسایل و دستگاه‌های پیرامون ما (خودپرداز، تلفن همراه، اتومبیل و ماشین لباسشویی) سیستم نهفته قرار دارد. شکل زیر یک سیستم امنیتی را در دو شکل نهفته و PC Based (مبتنی بر PC) نشان می‌دهد.



در علم کامپیوتر، محاسبات بی‌درنگ (RTC: Real –Time Computing) و یا محاسبات واکنشی (Reactive Commuting)، سیستم‌های نرم‌افزاری و سخت افزاری را توصیف می‌کند که زمان در آنها اهمیت دارد و بایستی این تضمین را حاصل کنند که خروجی صحیح سیستم، در یک بازه‌ی مشخصی حتما تولید شود، چرا که تولید این خروجی در خارج از این بازه زمانی، حتی اگر صحیح نیز باشد، دیگر مطلوب نخواهد بود. این آستانه‌های زمانی، سرحد زمانی یا Deadlne نامیده می‌شود. پاسخ‌های سیستم‌های بی‌درنگ به رویدادها اغلب در حدود میلی ثانیه و میکروثانیه هستند.

همانطور که گفتیم سیستم‌های بی‌درنگ در عمل بر دو نوع تقسیم‌بندی می‌شوند. یکی سیستم‌های بی‌درنگ نرم (Soft) و دیگری سیستم‌های بی‌درنگ سخت (Hard) که در شکل زیر قابل مشاهده است:



در یک سیستم **بی‌درنگ سخت**، زمان پاسخ به رویداد هم در حالت میانگین و هم در بدترین حالت هر دو کمتر از زمان Deadline هستند، اما در سیستم‌های **بی‌درنگ نرم**، زمان پاسخ به رویداد فقط در حالت میانگین کمتر از زمان Deadline است، و در بدترین حالت حتی ممکن است بیشتر از زمان Deadline باشد و از Deadline هم عبور کند. در حقیقت هنگام طراحی یک سیستم بی‌درنگ از نوع سخت باید دقیق و سخت‌گیرانه عمل کرد، اما در طراحی سیستم بی‌درنگ از نوع نرم نیازی به این سخت‌گیری زیاد نمی‌باشد.

مفهوم سیستم بی‌درنگ نهفته (Real Time Embedded Systems)، یک مفهوم علمی است که علوم متفاوتی را الزاما در بر می‌گیرد. این سیستم‌ها، همان طور که از نامش پیداست، تلفیقی از دو سیستم نهفته و بی‌درنگ است. یک سیستم نهفته بی‌درنگ را معمولا اینگونه تعریف می‌کنند، «سیستمی که محیط را از طریق دریافت داده‌ها، پردازش آنها و سپس برگرداندن سریع تاثیر پردازش این داده‌ها به محیط، کنترل می‌کند.»

توجه: در سیستم عامل‌های عمومی و همه منظوره یا GPOS (General Purpose Operating System) کاربر هیچ گونه کنترلی بر تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها ندارد، ولی در سیستم عامل‌های بی‌درنگ یا RTOS (Real Time Operating System) به خصوص بی‌درنگ سخت لازم است به کاربر به طور وسیع و گسترده اجازه تعیین **اولویت‌ها، مهلت‌ها** و **کنترل زمان‌بندی فرآیندها و نخ‌ها** داده شود.

صورت سوال به این شکل است:

در سیستم‌های تعبیه شده بی‌درنگ سخت که پاسخ در زمانی مشخص باید تضمین شود، کدام روش نگاشت ریسمان‌های کاربر به ریسمان‌های سیستمی، مناسب است؟

۱) یک به یک

گزینه اول پاسخ سوال نیست، زیرا در روش یک به یک فقط زمان‌بند پردازنده و زمان‌بند چندنخی در سطح هسته وجود دارد و زمان‌بند چند نخی در سطح کاربر در این روش مورد

استفاده قرار نمی‌گیرد. در واقع هسته سیستم عامل نه تنها فرآیندها را می‌شناسد بلکه از وجود نخ-های داخل یک فرآیند نیز در صورت وجود نخ آگاه است. در واقع مدیریت نخ‌ها و زمان‌بند چندنخی (نخ‌های فرآیندهای کاربر و نخ‌های فرآیندهای سیستم عامل) در سطح هسته، توسط هسته سیستم عامل انجام می‌گردد و کاربر هیچ دیدی از این کار ندارد. در سیستم‌های بی‌درنگ به خصوص بی‌درنگ سخت لازم است به کاربر به طور وسیع و گسترده اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها داده شود. در روش یک به یک، سطح هسته وجود دارد و به تبع همروندی (در سیستم‌های تک‌پردازنده‌ای) پس از مسدود شدن یک نخ و توازی (در سیستم‌های چندپردازنده‌ای) وجود دارد که این امر منجر به این می‌شود که زمان پاسخ کوتاه باشد و پاسخ قبل از اتمام مهلت زمانی تولید شود. اما در روش یک به یک، سطح کاربر وجود ندارد، که این امر منجر به این می‌شود که اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها به طور وسیع و گسترده به کاربر داده نشود.

۲) چند به یک

گزینه دوم پاسخ سوال نیست، زیرا در روش چند به یک فقط زمان‌بند پردازنده و زمان‌بند چند نخی در سطح کاربر وجود دارد و زمان‌بند چند نخی در سطح هسته در این روش مورد استفاده قرار نمی‌گیرد. در واقع هسته سیستم عامل فقط فرآیندها را می‌شناسد و هیچ اطلاعاتی از نخ‌ها ندارد. در واقع اولویت‌بندی نخ‌ها، مدیریت نخ‌ها و زمان‌بند چندنخی در سطح کاربر و توسط یک بسته نرم‌افزاری انجام می‌گردد. بدین معنی که نخ‌ها را برنامه‌نویس مشخص می‌کند و مدیریت آن-ها را نیز بر عهده می‌گیرد. در سیستم‌های بی‌درنگ به خصوص بی‌درنگ سخت لازم است به کاربر به طور وسیع و گسترده اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها داده شود. در روش چند به یک، سطح هسته وجود ندارد و به تبع همروندی (در سیستم‌های تک-پردازنده‌ای) پس از مسدود شدن یک نخ و توازی (در سیستم‌های چندپردازنده‌ای) وجود ندارد که این امر منجر به این می‌شود که زمان پاسخ طولانی شود و به تبع این احتمال وجود دارد که پاسخ بعد از اتمام مهلت زمانی تولید شود. هرچند که در روش چند به یک، سطح کاربر وجود دارد، که این امر منجر به این می‌شود که اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها به طور وسیع و گسترده به کاربر داده شود.

۳) چند به چند

گزینه سوم پاسخ سوال نیست، زیرا روش چند به چند از اجتماع دو روش سطح کاربر و هسته ابداع گردیده است. در روش چند به چند علاوه بر زمان‌بند پردازنده و زمان‌بند چندنخی در سطح هسته سیستم عامل، زمان‌بند چند نخی نیز در سطح کاربر برای زمان‌بندی و اولویت‌دهی نخ‌های فرآیند کاربر وجود دارد. در واقع زمان‌بندی نخ‌های فرآیندهای سیستم عامل، توسط هسته سیستم عامل و زمان‌بندی نخ‌های فرآیندهای کاربر، توسط برنامه کاربر انجام می‌گردد، که این امر منجر به اولویت‌بندی نخ‌های فرآیندهای کاربر می‌گردد. در این روش، زمان‌بند چند نخی در سطح کاربر،

نخ‌های کاندید خود را از میان نخ‌های متعدد در فرآیندهای مختلف کاربر براساس یک الگوریتم خاص انتخاب و تحویل یک زمان‌بند چند نخی در سطح هسته می‌دهد. در واقع انتخاب نخ‌های کاندید (زمان‌بندی) در فرآیندهای کاربر به خود کاربر واگذار شده است که همانطور که گفتیم منجر به اولویت‌بندی نخ‌های فرآیندهای کاربر می‌گردد.

در سیستم‌های بی‌درنگ به خصوص بی‌درنگ سخت لازم است به کاربر به طور وسیع و گسترده اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها داده شود. در روش چند به چند، سطح هسته وجود دارد و به تبع همروندی (در سیستم‌های تک‌پردازنده‌ای) پس از مسدود شدن یک نخ و توازی (در سیستم‌های چندپردازنده‌ای) وجود دارد که این امر منجر به این می‌شود که زمان پاسخ کوتاه باشد و پاسخ قبل از اتمام مهلت زمانی تولید شود. اما یک ریسک دارد اینکه در روش چند به چند برنامه‌نویس به تعداد دلخواه نخ ایجاد می‌کند، اما ممکن است تعداد LWP کمتری به آنها منتسب شود تا سربار هسته کمتر شود. زیرا در روش چند به چند نخ‌های سطح کاربر به تعداد کم‌تر یا مساوی از نخ‌های سطح هسته نگاشت می‌شود. برای مثال اگر یک فرآیند پنج درخواست(نخ) همزمان داشته باشد، اگر چهار LWP داشته باشیم، یکی از درخواست‌ها(نخ‌ها) تا بازگشت نتیجه یکی از چهار درخواست(نخ) دیگر از سطح هسته به جریان نخواهد افتاد و این یعنی تاخیر و ممکن است زمان پاسخ طولانی شود و به تبع این احتمال وجود دارد که پاسخ بعد از اتمام مهلت زمانی تولید شود. هرچند که در روش چند به چند، سطح کاربر وجود دارد، که این امر منجر به این می‌شود که اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها به طور وسیع و گسترده به کاربر داده شود.

۴) دوسطحی

گزینه چهارم پاسخ سوال است، زیرا روش دو سطحی به نوعی روش **چند به چند پلاس** است، یعنی روش دوسطحی بهبود یافته روش چند به چند است، در واقع روش دوسطحی علاوه بر اینکه روش چند به چند را در خود دارد، جهت بهبود کارایی روش یک به یک را نیز در خود قرار داده است. در سیستم‌های بی‌درنگ به خصوص بی‌درنگ سخت لازم است به کاربر به طور وسیع و گسترده اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها داده شود. در روش دوسطحی (Two-Level Model) یا همان چند به چند پلاس، سطح هسته وجود دارد و به تبع همروندی (در سیستم‌های تک‌پردازنده‌ای) پس از مسدود شدن یک نخ و توازی (در سیستم-های چندپردازنده‌ای) وجود دارد که این امر منجر به این می‌شود که زمان پاسخ کوتاه باشد و پاسخ قبل از اتمام مهلت زمانی تولید شود. اما یک ریسک دارد اینکه در روش چند به چند برنامه‌نویس به تعداد دلخواه نخ ایجاد می‌کند، اما ممکن است تعداد LWP کمتری به آنها منتسب شود تا سربار هسته کمتر شود. زیرا در روش چند به چند نخ‌های سطح کاربر به تعداد کم‌تر یا مساوی از نخ‌های سطح هسته نگاشت می‌شود. برای مثال اگر یک فرآیند پنج درخواست(نخ) همزمان داشته باشد، اگر چهار LWP داشته باشیم، یکی از درخواست‌ها(نخ‌ها) تا بازگشت نتیجه

یکی از چهار درخواست(نخ) دیگر از سطح هسته به جریان نخواهد افتاد و این یعنی تاخیر و ممکن است زمان پاسخ طولانی شود و به تبع این احتمال وجود دارد که پاسخ بعد از اتمام مهلت زمانی تولید شود. اما از آنجا که روش دوسطحی علاوه بر داشتن روش چند به چند، روش یک به یک را نیز دارد، می‌تواند در مواقع لزوم برای پاسخ سریع از روش یک به یک نیز استفاده نماید، زیرا در روش یک به یک هر رشته نخ کاربر به یک رشته نخ هسته نگاست می‌شود. همچنین در روش دوسطحی (Two-Level Model) یا همان چند به چند پلاس، سطح کاربر وجود دارد، که این امر منجر به این می‌شود که اجازه تعیین اولویت‌ها، مهلت‌ها و کنترل زمان‌بندی فرآیندها و نخ‌ها به طور وسیع و گسترده به کاربر داده شود. روش دوسطحی در سیستم عامل‌هایی نظیر IRIX، HP-UX و True64nIx پشتیبانی می‌شود. نسخه‌های قبل از Solaris9 نیز مدل دوسطحی را پشتیبانی می‌کنند، اما Solaris9 از روش یک به یک استفاده می‌کند.

تست‌های فصل هشتم

۹۸- یک دیسک سخت متشکل از 200 شیار که از 0 تا 199 شماره‌گذاری شده است و سر بازوی آن روی شیار 50 و در جهت افزایش شیار است را در نظر بگیرید. اگر درخواست پیمایش شیارهای زیر به ترتیب (از چپ به راست) باشد، و الگوریتم C-LOOK برای زمان‌بندی استفاده شده باشد، تعداد کل شیار پیمایش شده کدام است؟

95,180,34,119,11,123,62,64
(مهندسی *IT* – دولتی ۹۷)

۱) 299
۲) 322
۳) 337
۴) 382

پاسخ‌های فصل هشتم

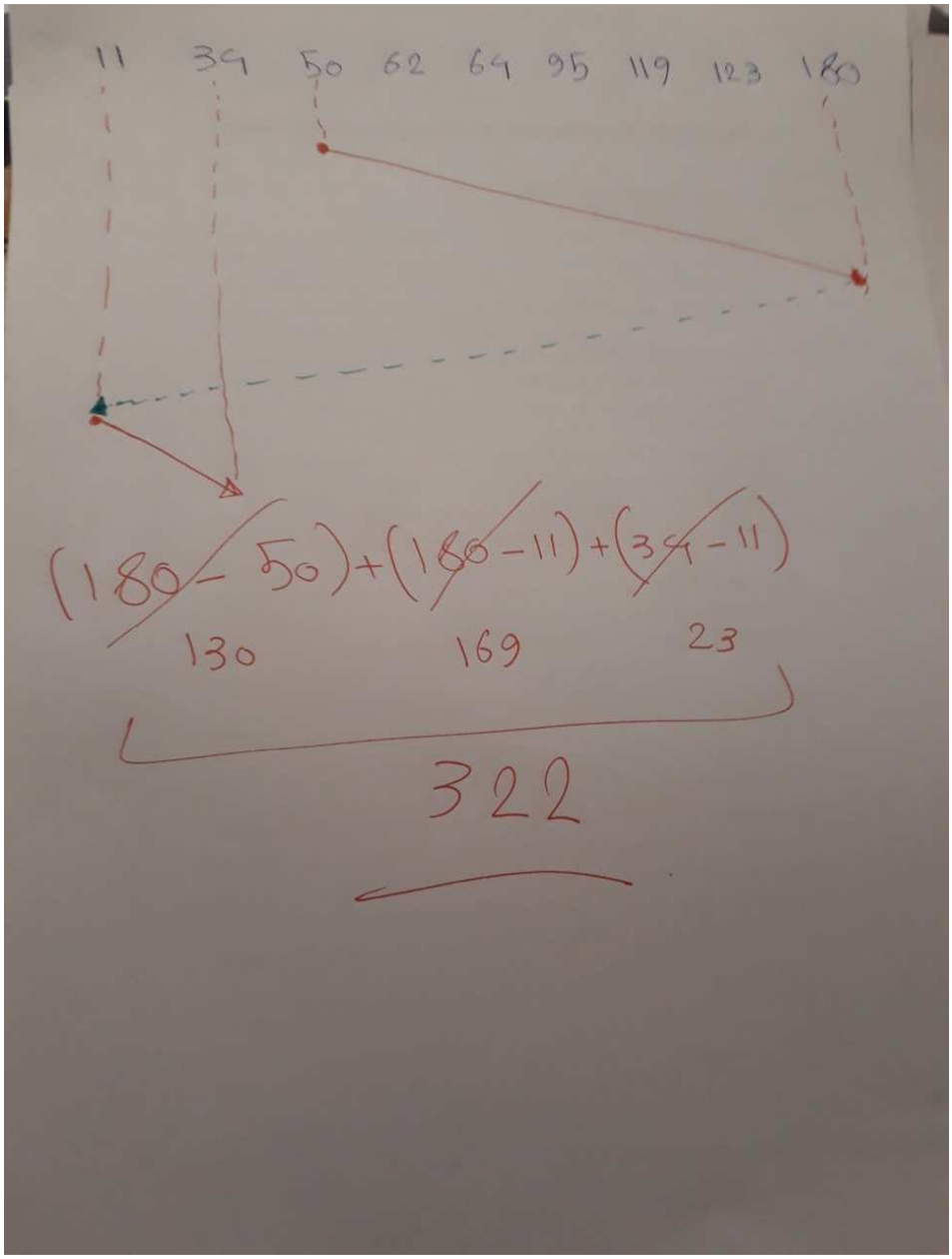
۹۸- گزینه (۲) صحیح است.

الگوریتم C-LOOK همانند روش LOOK می‌باشد، اما هد فقط در یک جهت حرکت به درخواست‌ها سرویس می‌دهد (در واقع در مسیر برگشت به هیچ درخواستی سرویس نمی‌دهد). در این روش، هد در یک جهت درخواست‌ها را سرویس داده و وقتی به آخرین درخواست رسید، سریعاً به اولین درخواست دیسک برگشته و دوباره در همان جهت شروع به سرویس دادن می‌کند. با این روش نیازهای درونی، بیرونی و میانی به یک اندازه انتظار می‌کشند.

توجه: مطابق صورت سؤال، هد روی شیار 50 دیسک قرار دارد و به سمت شیار 199 یعنی در جهت افزایش شیار مطابق فرض سوال در حال حرکت است. بنابراین داریم:

$$50 \xrightarrow{12} 62 \xrightarrow{2} 64 \xrightarrow{31} 95 \xrightarrow{24} 119 \xrightarrow{4} 123 \xrightarrow{57} 180 \xrightarrow{169} 11 \xrightarrow{23} 34$$

مجموع جابجایی بین شیارها با زمان‌بندی C-LOOK: $(180-50)+(180-11)+(34-11)=130+169+23=322$



تست‌های فصل هفتم

۹۹- وضعیت موجود در یک سیستم به‌صورت جدول زیر است. در دو حالت ۱) وضعیت جاری و ۲) وضعیتی که فرآیند P1 درخواست منابع بیشتر به صورت (A=0, B=5, C=2) بدهد، کدام مورد از نظر امن بودن به ترتیب (از راست به چپ، درست است؟

	Max	Allocation	Available
	A B C	A B C	A B C
P_0	0 0 1	0 0 1	
P_1	1 7 5	1 0 0	
P_2	2 3 5	1 3 5	
P_3	0 6 5	0 6 3	
Total		2 9 9	1 5 2

۱) امن، امن ۲) امن، ناامن ۳) ناامن، امن ۴) ناامن، ناامن

ارسطو خلیلی‌فر (مولف کتاب سیستم عامل راهیان ارشد)

سوال ۹۹ کنکور مهندسی IT – دولتی ۱۳۹۷

درجه اهمیت سوال: بسیار بالا

ناشر: گروه بابان و راهیان ارشد

پاسخ‌های فصل هفتم

۹۹- گزینه (۱) صحیح است.

ابتدا وضعیت سیستم را قبل از اجابت درخواست فرآیند P_1 بررسی می‌کنیم:

با توجه به ماتریس تخصیص منابع (Allocation) و منابع در دسترس یعنی بردار Available، منابع اولیه را بدست می‌آوریم، برای یافتن منابع اولیه ابتدا برای هر منبع، منابع تخصیص یافته به هر فرآیند را با هم جمع کرده و سپس با منابع در دسترس یعنی بردار Available جمع می‌نماییم.

A منبع اولیه $= 1+(1+1) = 3$

B منبع اولیه $= 5+(3+6) = 14$

C منبع اولیه $= 2+(1+5+3) = 11$

بنابراین منابع اولیه به صورت زیر است:

منابع اولیه	3	14	11

براساس رابطه زیر داریم:

$$Need = MAX - Allocation$$

فرآیند	Need		
	A	B	C
P_0	0	0	0
P_1	0	7	5
P_2	1	0	0
P_3	0	0	2

=

فرآیند	MAX		
	A	B	C
P_0	0	0	1
P_1	1	7	5
P_2	2	3	5
P_3	0	6	5

−

فرآیند	Allocation		
	A	B	C
P_0	0	0	1
P_1	1	0	0
P_2	1	3	5
P_3	0	6	3

Available	1	5	2

$$(1,5,2) \xrightarrow[+(0,0,1)]{P_0} (1,5,3) \xrightarrow[+(0,6,3)]{P_3} (1,11,6) \xrightarrow[+(1,3,5)]{P_2} (2,14,11)$$

$$\xrightarrow[+(1,0,0)]{P_1} (3,14,11)$$

به این ترتیب و با توجه به ماتریس‌های فوق، دنباله‌ی امنِ $< P_0, P_3, P_2, P_1 >$ (از چپ به راست) برای این سیستم موجود است. بنابراین قبل از اجابت درخواست فرآیند P_1، سیستم در حالت امن قرار دارد و گزینه‌های سوم و چهارم نادرست هستند.

الگوریتم درخواست منبع بانکداران

هنگامی که فرآیند P_i، بردار Request[i] را به عنوان اعلام نیاز به منابع مطرح می‌کند، باید این درخواست مطابق مراحل زیر بررسی گردد:

۱) اگر $Request[i] \leq Need[i]$ است به مرحله بعدی برو، در غیر اینصورت خطای تقاضای بیش از نیاز اعلام می‌شود. در واقع در این مرحله بررسی می‌شود که اگر فرآیند درخواست بیش از آن چه که در ابتدا اعلام نیاز کرده است را دارد، با این درخواست مخالفت شود.

۲) اگر $Request[i] \leq Availabe[i]$ است به مرحله 3 برو، در غیر اینصورت منبع کافی جهت تخصیص به این فرآیند وجود نداشته و این فرآیند باید منتظر بماند.

۳) با فرض این که منابع مورد نیاز به این فرآیند اختصاص می‌یابد، در این صورت با اصلاح ماتریس‌ها و بردارها یک الگوریتم تشخیص وضعیت امن اجرا می‌شود. با اجرای این الگوریتم اگر سیستم در وضعیت امن باقی بماند، این منابع به فرآیند اختصاص می‌یابند، در غیراینصورت اگر با این تخصیص سیستم به وضعیت ناامن وارد شود، ماتریس‌ها و بردارها به حالت قبل بازگردانده شده و فرآیند تا آزاد شدن منابع بیشتر منتظر می‌ماند.

اصلاح ماتریس‌ها و بردارها در صورت تخصیص به صورت زیر انجام می‌شوند:

Available (new) = Available (old) – Request[i]

Need (new) [i] = Need (old) [i] – Request[i]

Allocation (new) [i] = Allocation (old) [i] + Request[i]

مطابق فرض صورت سؤال، اگر در این وضعیت، درخواستی برای صفر واحد دیگر از منبع A، پنج واحد دیگر از منبع B و دو واحد دیگر از منبع C توسط فرآیند P_1 صادر شود، مطابق الگوریتم درخواست منبع بانکداران، ابتدا بررسی می‌شود که آیا درخواست فرآیند، کم‌تر از نیاز اعلام شده آن است یا خیر.

$Request[P_1] \leq Need[P_1]$

$[0,5,2] \leq [0,7,5]$

چون شرط اول برقرار است، به سراغ شرط دوم می‌رویم.

در شرط دوم بررسی می‌شود که آیا درخواست داده شده، کمتر از منابع آزاد سیستم است یا خیر.

$Request[P_1] \leq Available$

$[0,5,2] \leq [1,5,2]$

چون هر دو شرط برقرار است، سراغ مرحله سوم از الگوریتم می‌رویم.

در این مرحله، با فرض اینکه درخواست داده شده، اعمال شود، ماتریس‌ها به صورت زیر بروزرسانی می‌شوند:

فرآیند	Need (new)		
	A	B	C
P_0	0	0	0
P_1	0	②	③
P_2	1	0	0
P_3	0	0	2

=

فرآیند	MAX		
	A	B	C
P_0	0	0	1
P_1	1	7	5
P_2	2	3	5
P_3	0	6	5

−

فرآیند	Allocation (new)		
	A	B	C
P_0	0	0	1
P_1	1	⑤	②
P_2	1	3	5
P_3	0	6	3

با توجه به ماتریس تخصیص منابع **جدید** ((Allocation(new)) و منابع اولیه، بردار Available(new) را بدست می‌آوریم، برای یافتن بردار Available(new) ابتدا برای هر منبع، منابع تخصیص‌یافته به هر فرآیند را با هم جمع کرده و سپس از منابع اولیه کسر می‌نماییم.

A منبع موجود $= 3-(1+1) = 1$

B منبع موجود $= 14-(5+3+6) = 0$

C منبع موجود $= 11-(1+2+5+3) = 0$

بنابراین بردار Available(new) به صورت زیر است:

Available(new)	①	⓪	⓪

مطابق رابطه زیر نیز می‌توان Available (new) را محاسبه نمود:

Available (new) = Available (old) – Request $[P_1]$

Available (new) = [1,5,2] – [0,5,2] = [1,0,0]

$$(1,0,0) \xrightarrow[+(0,0,1)]{P_0} (1,0,1) \xrightarrow[+(1,3,5)]{P_2} (2,3,6) \xrightarrow[+(1,5,2)]{P_1} (3,8,8)$$

$$\xrightarrow[+(0,6,3)]{P_3} (3,14,11)$$

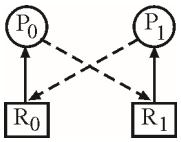
به این ترتیب و با توجه به ماتریس‌های فوق، دنباله‌ی امنِ $< P_0, P_2, P_1, P_3 >$ (از چپ به راست) برای این سیستم موجود است. بنابراین بعد از اجابت درخواست فرآیند P_1، سیستم بازهم در حالت امن قرار دارد و گزینه دوم نیز نادرست و به تبع گزینه‌ی اول درست و پاسخ سوال است.

توجه: ناامنی نه به معنی وقوع بن‌بست در گذشته و نه به معنی وقوع حتمی بن‌بست در آینده است، بلکه به معنی احتمال وقوع بن‌بست در آینده است.

توجه: در حالت ناامن نیز ممکن است، فرآیندها در موقعیت رهاسازی منابع در اختیار خود قرار بگیرند و سیستم به بن‌بست نرسد. به بیان دیگر در حالت ناامن وقوع بن‌بست قطعی نیست. اما اگر در حالت ناامن فرآیندها علاوه بر نگهداری منابع تحت تملک خود، منابع دیگری را مورد درخواست قرار دهند، در این حالت فرآیندهایی که نیاز آن‌ها برطرف نمی‌شود یک به یک در صف انتظار قرار می‌گیرند و همچون سرنوشت‌های به هم گره خورده، تا ابد منتظر یکدیگر باقی می‌مانند، یعنی بن‌بست رخ داده است.

مثال: در یک سیستم تعداد منبع اولیه R_0 برابر یک و تعداد منبع اولیه R_1 برابر یک است. اگر فرآیند P_0 یک منبع R_0 را در تملک داشته باشد و برای اتمام، نیاز به یک منبع R_1 نیز داشته باشد و همچنین فرآیند P_1 یک منبع R_1 را در تملک داشته باشد و برای اتمام، نیاز به یک منبع R_0 نیز داشته باشد، وضعیت سیستم را در این شرایط بررسی کنید:

حل: ماتریس و گراف منابع و فرآیندها به صورت زیر است:



فرآیند	MAX	
	R_0	R_1
P_0	1	1
P_1	1	1

=

فرآیند	Allocation	
	R_0	R_1
P_0	1	0
P_1	0	1

+

فرآیند	Need	
	R_0	R_1
P_0	0	1
P_1	1	0

با توجه به ماتریس تخصیص منابع (Allocation) و منابع اولیه، بردار Available را بدست می‌آوریم:

R_0 منبع موجود $= 1-(1) = 0$

R_1 منبع موجود $= 1-(1) = 0$

بنابراین بردار Available به صورت زیر است:

Available	0	0

مشاهده می‌شود که با بردار Available موجود، نمی‌توان نیاز هیچ فرآیندی را با توجه به ماتریس Need مرتفع ساخت، بنابراین توالی امن یافت نمی‌شود و سیستم در یک وضعیت ناامن قرار دارد و احتمال وقوع بن‌بست وجود دارد.

توجه: در این شرایط ناامن، اگر فرآیند P_0 یا P_1 و یا هر دو، در موقعیت رهاسازی منابع در اختیار خود قرار بگیرند، یعنی رهاسازی کنند و سپس درخواست منابع باقی‌مانده خود را صادر کنند، بن‌بست رخ نمی‌دهد.

توجه: در این شرایط ناامن، اگر فرآیند P_0 در عین حال اینکه منبع R_0 را تحت تملک و نگهداری خود دارد، منبع R_1 را نیز درخواست کند، از آن‌جا که منبع R_1 در اختیار فرآیند P_1 می‌باشد، فرآیند P_0 در صف انتظار، می‌خوابد (نگهداری و انتظار) حال اگر در ادامه ماجرا، فرآیند P_1 نیز همین رویه را در پیش گیرد، یعنی فرآیند P_1 نیز در عین حال اینکه منبع R_1 را تحت تملک و نگهداری خود دارد، منبع R_0 را نیز درخواست کند، از آن‌جا که منبع R_0 در اختیار فرآیند P_0 می‌باشد، فرآیند

P_1 نیز در صف انتظار، می‌خوابد (نگهداری و انتظار). این سرنوشت‌های بـه هـم گـره خـورده، در ادامه بن‌بست را به ارمغان می‌آورد.

توجه: از دو توجه فوق این نتیجه استنباط می‌گردد که در شرایط ناامن، **احتمـال وقـوع بن‌بسـت** وجود دارد. در واقع در شرایط ناامن وقوع بن‌بست و عدم بن‌بسـت بـه رفتـار فرآینـدها در آینـده، بستگی دارد. یعنی فرآیندها رفتار نگهداری و درخواست را در پیش گیرند و یا رفتـار رهاسـازی و درخواست را.

تست‌های فصل چهارم

۱۰۰- حافظه‌ای با بخش‌های (partitions) ۱۰۰ ، ۴۰۰ ، ۲۰۰ و ۵۰۰ کیلوبایت را در نظر بگیرید. اگر از روش بدترین تطابق (Worst Fit) برای اختصاص فضای حافظه به فرایندها استفاده شود، مشخص کنید فرایندهای با اندازه‌های زیر به ترتیب (از راست به چپ) در کدام بخش از حافظه قرار می‌گیرد؟

فرآیند	P1	P2	P3	P4
اندازه (کیلوبایت)	۲۰۰	۳۹۵	۱۰۰	۲۵۴

۱) ۴۰۰-۱۰۰-۵۰۰-۲۰۰

۲) ۵۰۰-۱۰۰-۴۰۰-۲۰۰

۳) ۵۰۰-۴۰۰-۵۰۰- فضا وجود ندارد

۴) ۵۰۰-۴۰۰-۲۰۰- فضا وجود ندارد

پاسخ‌های فصل چهارم

۱۰۰- گزینه (۳) صحیح است.

روش Worst Fit

در این روش باید کل فضای حافظه جست‌وجو شود تا همیشه بزرگترین فضای موجود را به هر فرآیند تخصیص دهیم!

در واقع ایده این روش این است که پس از تخصیص بزرگترین حفره به یک فرآیند، فضای باقیمانده آنقدر بزرگ هست که باز هم بتوان از آن استفاده کرد. در صورتی که در Best Fit، پس از آن‌که یک حفره به یک فرآیند اختصاص داده شد، فضای باقیمانده آنقدر کوچک است که بعدها به کار هیچ فرآیندی نمی‌آید.

توجه: در روش worst fit اولین پیشنهاد بزرگ برای یک فرآیند آخرین پیشنهاد هم هست چون گزینه بزرگتر دیگری ندارد و بزرگترینی که می‌توانست پیشنهاد دهد، همان اول پیشنهاد داد.

توجه: روش‌های Best Fit و Worst Fit نسبت به روش‌های Next Fit و First Fit قدری کندتر هستند، زیرا در این دو روش تمام حافظه باید جست‌وجو شود.

توجه: در روش Best Fit حافظه پر از حفره‌های بسیار کوچکی می‌شود که به هیچ کار نمی‌آیند. در روش Worst Fit نیز ممکن است فرآیندهای بزرگ دچار گرسنگی شوند، زیرا قسمت‌های بزرگتر، زودتر تخصیص داده شده و کوچک می‌شوند.