

موسسه بابان

انتشارات بابان و انتشارات راهیان ارشد

درس و کنکور ارشد

سیستم عامل

(مدیریت فرآیندها و زمان بندی پردازنده)

ویژه‌ی داوطلبان کنکور کارشناسی ارشد مهندسی کامپیوتر و IT

بر اساس کتب مرجع

آبراهام سیلبرشاتز، ویلیام استالینگز و اندرو اس تنن‌بام

ارسطو خلیلی فر

کلیه‌ی حقوق مادی و معنوی این اثر در سازمان اسناد و کتابخانه‌ی ملی ایران به ثبت رسیده است.

مدیریت فرآیند

۲

مقدمه

تعاریف مختلفی برای فرآیند ارائه می‌شود، از جمله اینکه فرآیند، برنامه‌ی در حال اجراست. اما می‌دانیم فرآیند ممکن است همیشه در حال اجرا نباشد، مثلاً منتظر باشد. بنابراین به بیان تعریف دیگری می‌پردازیم:

از لحظه‌ای که یک کار توسط زمانبند کار انتخاب و وارد گردونه‌ی مراحل مختلف می‌شود، تا لحظه‌ای که اجرای آن به طور کامل خاتمه یافته و از سیستم خارج می‌شود، فرآیند (Process) نام دارد. **نکته:** یک برنامه به خودی خود فرآیند نیست. زیرا برنامه یک موجودیت غیر فعال (Passive) است که مثلاً در قالب یک فایل و بر روی دیسک ذخیره شده است. اما فرآیند یک موجودیت فعال (Active) می‌باشد که در حال اجراست.

نکته: یک فرآیند در واقع شامل اجزایی مانند رجیسترهای CPU (از جمله شمارنده‌ی برنامه (PC)، یک پشته در حافظه (جهت نگهداری داده‌های موقتی، متغیرهای محلی، پارامترهای توابع و زیرروال‌ها) و بخش داده‌ای (Data Segment) (شامل داده‌ها و متغیرهای سراسری) است. **توجه:** شمارنده برنامه یا PC (program counter)، آدرس دستورالعمل بعدی فرآیند را نشان می‌دهد. **نکته:** سیستم عامل اطلاعات مربوط به فرآیندها را در جدولی با عنوان Process Table درج و نگهداری می‌کند. در این جدول به هر فرآیند یک درایه (رکورد) اختصاص داده می‌شود که به آن PCB (Process Control Block) گویند.

۱	PCB
۲	PCB
۳	PCB
⋮	⋮
n	PCB

جدول فرآیند

نکته: PCB در واقع یک ساختمان داده جهت نگهداری اطلاعات مربوط به فرآیندها نزد سیستم عامل

است که در آن اطلاعاتی از جمله موارد زیر برای هر فرآیند ذخیره می شود:

- شناسه فرآیند (Process ID)
 - شناسه‌ی کاربر فرآیند یا حتی شناسه‌ی گروهی که کاربر فرآیند عضو آن است (UID,GID)
 - اولویت فرآیند
 - رجیسترهای CPU (شامل PC ، PSW و مابقی رجیسترها)
 - حالت و وضعیت فرآیند (مانند جدید، آماده، در حال اجرا، منتظر و معلق)
 - اطلاعاتی راجع به منابع در اختیار فرآیند (فایل‌ها، حافظه، پردازنده)
 - اطلاعاتی راجع به میزان مصرف منابع توسط فرآیند (مانند میزان پردازنده‌ی مصرفی)
 - اطلاعات مربوط به زمانبندی
 - اشاره‌گرهایی به قسمت کد، داده و پشته فرآیند (DS ، CS و SP)
 - اطلاعات مربوط به مدیریت حافظه (جداول صفحه، قطعه و مسائل حفاظتی)
- در واقع PCB مانند یک مخزن است که اطلاعاتی که از یک فرآیند به فرآیند دیگر متغیر است، در آن ذخیره می شود.
- توجه:** رجیستر PSW (program status word) : در این رجیستر مُد کاربر یا هسته مربوط به پردازنده نگهداری می شود. اگر صفر باشد، پردازنده در مُد هسته و اگر یک باشد، پردازنده در مُد کاربر قرار دارد.

سرگذشت فرآیند

یک فرآیند طی مدت زمانی که در سیستم وجود دارد، می تواند حالت‌های زیر را داشته باشد:

- حالت اول) جدید (New):** فرآیند در حال ایجاد شدن است.
- حالت دوم) آماده (Ready):** فرآیند درون حافظه قرار دارد و همه منابع مورد نیاز به جز CPU در اختیار آن است (در واقع منتظر است تا نوبت CPU به آن برسد).

حالت سوم) در حال اجرا (Running): فرآیند CPU را در اختیار دارد و در حال اجراست (بالتبع همه منابع موردنیاز دیگر نیز در اختیار آن می‌باشد).

حالت چهارم) در حال انتظار (Waiting): در این حالت فرآیند منتظر وقوع یک رویداد (Event) است (برای مثال منتظر اختصاص یک منبع (به جز CPU) یا تکمیل شدن یک عمل I/O). در واقع وقتی فرآیندی در حالت انتظار است، نیازی به CPU ندارد. به این حالت مسدود (Blocked) نیز می‌گویند. **حالت پنجم)** خاتمه یافته (Terminated): اجرای فرآیند پایان یافته است.

نکته: در حالت‌های جدید، آماده و انتظار، صفی برای فرآیندها وجود دارد، اما در حالت اجرا صف نداریم چون در هر لحظه فقط یک فرآیند پردازنده را در اختیار دارد و در حال اجراست.

تغییر حالات ممکن برای یک فرآیند

۱- از New به Ready

اگر سیستم عامل آماده دریافت یک فرآیند باشد، فرآیند از حالت جدید به حالت آماده منتقل می‌شود. این انتقال توسط زمانبند کار (Job Scheduler) انجام می‌شود.

۲- از Ready به Running

در این حالت پردازنده به یک فرآیند تخصیص داده می‌شود تا اجرا گردد. این کار توسط زمانبند فرآیند (Process Scheduler) انجام می‌شود.

۳- از Running به Ready

در این حالت پردازنده از فرآیند گرفته شده و فرآیند دوباره به صف فرآیندهای آماده منتقل می‌شود. **نکته:** CPU به یکی از دلایل زیر می‌تواند از یک فرآیند گرفته شود:
- فرآیند به طور اختیاری پردازنده را رها کند (مثلاً به I/O نیاز پیدا می‌کند).
- سهمیه فعلی اجرای فرآیند به پایان برسد.
- یک فرآیند با اولویت بالاتر به پردازنده نیاز داشته باشد.

۴- از Running به (Blocked) Waiting

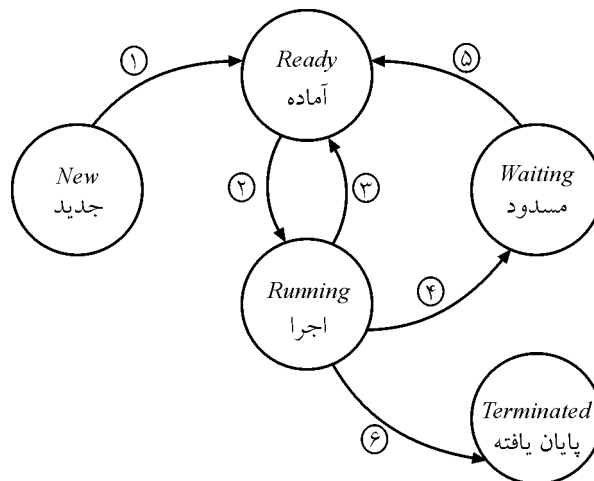
این تغییر حالت وقتی رخ می‌دهد که یک فرآیند در حال اجرا، منتظر وقوع یک رویداد است، برای مثال به I/O نیاز پیدا می‌کند.

۵- از (Blocked) Waiting به Ready

وقتی رویدادی که فرآیند منتظر آن است (و به خاطر آن مسدود شده است) رخ می‌دهد، فرآیند از حالت مسدود به آماده می‌رود.

۶- از Running به Terminated

وقتی فرآیندی به اتمام برسد، از حالت اجرا به پایان یافته منتقل می شود.
نکته: تغییر حالات ممکن برای یک فرآیند در شکل زیر نشان داده شده است (شماره‌های روی پیکان‌ها نشان دهنده تغییر حالت‌های مذکور می باشند):



تغییر حالت فرآیند - نمودار ۵ حالت

نکته: بیشتر سیستم عامل‌ها حالت‌های فرآیند را فقط محدود به پنج حالت جدید، اجرا، آماده، مسدود و پایان یافته نمی‌کنند، بلکه یک موقعیت جدید با عنوان معلق (Suspend) نیز برای فرآیند در نظر می‌گیرند. در حالت معلق فرآیند از حافظه به روی دیسک بیرون رانده شده است. به عنوان مثال فرض کنید همه فرآیندهای موجود در سیستم به حالت مسدود نقل مکان کنند (و همگی مثلاً منتظر اتمام ورودی - خروجی مختص به خود هستند)، بنابراین هیچ فرآیندی در صف آماده قرار ندارد. در این حالت سیستم عامل جهت بیکار نماندن CPU، می‌تواند یک فرآیند را که در حالت مسدود است، از حافظه به دیسک منتقل کرده و یک فرآیند جدید را وارد صف آماده کند.

نکته: با این توضیحات می‌توان دو حالت جدید برای فرآیند در نظر گرفت:

حالت ششم) منتظر و معلق (Suspend Wait): یک فرآیند که در حالت منتظر (مسدود) قرار دارد از حافظه به دیسک منتقل می‌شود. در واقع در این حالت منبع حافظه نیز از فرآیند گرفته شده است. به این حالت مسدود و معلق (Suspend Blocked) نیز می‌گویند.

حالت هفتم) آماده و معلق (Suspend Ready): یک فرآیند که بر روی دیسک است، به محض برگشتن به حافظه، می‌تواند اجرا شود. یک فرآیند در دو صورت در این حالت به سر می‌برد:

الف - به علت کمبود حافظه یک فرآیند مستقیماً از حالت آماده، به روی دیسک برده شود.

ب - یک فرآیند که بر روی دیسک است و در حالت مسدود و معلق به سر می‌برد، حادثه موردنظرش

رخ دهد (مانند اتمام I/O).

با در نظر گرفتن حالات ۶ و ۷، تغییر حالت‌های دیگری را نیز می‌توان برای یک فرآیند در نظر گرفت.

۷- از (Blocked) Waiting به (Suspend Blocked) Suspend Wait

هنگامی که هیچ فرآیندی، در صف آماده وجود نداشته باشد و همه فرآیندها مسدود (منتظر) باشند، سیستم عامل یک فرآیند را از صف مسدود انتخاب کرده و به روی دیسک منتقل می‌کند تا فضای موردنیاز برای فرآیند دیگری (جدید یا در حالت آماده و معلق) مهیا شود.

۸- از (Suspend Blocked) Suspend Wait به Suspend Ready

وقتی رویدادی که یک فرآیند مسدود و معلق منتظر آن بوده، رخ دهد (مانند تکمیل I/O) این فرآیند از حالت مسدود و معلق به حالت آماده و معلق منتقل می‌شود (دقت کنید فرآیند هنوز بر روی دیسک قرار دارد).

۹- از Suspend Ready به Ready

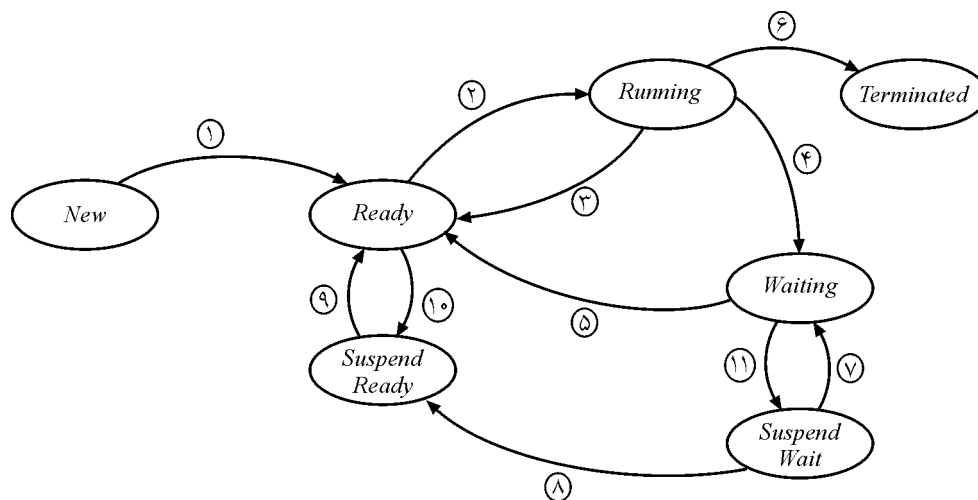
وقتی هیچ فرآیند آماده‌ای در سیستم موجود نباشد، اگر فرآیندی در حالت معلق و آماده وجود داشته باشد، سیستم عامل آن را به حافظه منتقل می‌کند. از طرفی اگر اولویت فرآیندی که در حالت آماده و معلق قرار دارد از همه فرآیندهای آماده، بالاتر باشد، سیستم عامل این فرآیند را به حافظه منتقل کرده و به حالت آماده (و احتمالاً بعد از آن به حالت اجرا) می‌برد.

۱۰- از Ready به Suspend Ready

احتمال وقوع این تغییر حالت بسیار کم است اما اگر فرآیندی که در حال اجراست به فضای حافظه بیشتری نیاز داشته باشد، سیستم عامل جهت خالی شدن حافظه باید یک فرآیند را به دیسک منتقل کند، حال اگر هیچ فرآیندی در حالت مسدود نباشد، یک فرآیند آماده، برای این کار انتخاب و به دیسک منتقل می‌شود.

۱۱- از (Suspend Blocked) Suspend Wait به (Blocked) Waiting

اگر حافظه خالی شده و فرآیندی در حالت مسدود و معلق وجود داشته باشد، سیستم عامل فرآیندهای مسدود و معلق را که بر روی دیسک هستند به حافظه منتقل می‌کند. با این توضیحات، نمودار تغییر حالات یک فرآیند به صورت زیر است. (شماره‌های روی پیکان‌ها نشان‌دهنده تغییر حالت‌های مذکور می‌باشند):



تغییر حالت فرآیند - نمودار ۷ حالت

زمانبندها

با توجه به تغییر حالات فرآیندها، زمانبندها از یک دیدگاه به ۳ دسته تقسیم می‌شوند:

۱- زمانبند بلند مدت (Long Term Scheduler)

این زمانبند درجه چند برنامه‌ریزی را مشخص می‌کند. در واقع از بین کارها، تعدادی را انتخاب کرده و به فرآیند تبدیل می‌کند. به این زمانبند، زمانبند کار (Job Scheduler) نیز می‌گویند.

۲- زمانبند میان مدت (Middle Term Scheduler)

این زمانبند بنا به دلایلی ممکن است فرآیندهایی را از روی حافظه، به دیسک و بالعکس منتقل کند. در واقع وظیفه این زمانبند جابه‌جا کردن فرآیندها بین حافظه و دیسک می‌باشد. به این کار مبادله (Swapping) گویند.

۳- زمانبند کوتاه مدت (Short Term Scheduler)

این زمانبند از بین فرآیندهای آماده در حافظه، یکی را جهت اجرا توسط پردازنده انتخاب می‌کند. از این رو به آن زمانبند فرآیند (Process Scheduler) نیز می‌گویند.

نکته: در کتاب پروفیسور تنن باوم

به زمانبند بلند مدت، زمانبند پذیرش (Admission Scheduler)،

به زمانبند میان مدت، زمانبند حافظه (Memory Scheduler) و

به زمانبند کوتاه مدت، زمانبند پردازنده (CPU Scheduler) نیز اطلاق می‌شود.

نکته: زمانبند بلند مدت باید انتخاب‌های خود را بسیار دقیق و سنجیده انجام دهد. برای مثال باید

همواره ترکیب مناسبی از فرآیندهای CPU Limited و I/O Limited را به سیستم وارد کند تا تعادل برقرار باشد.

نکته: سوئیچ کردن پردازنده بین فرآیندهای مختلف، تعویض متن (Context Switch) نام دارد. این عمل نیازمند ذخیره کردن حالت فرآیند قبلی و بارگذاری حالت فرآیند فعلی است. از آن جا که از نظر اجرایی، کار مفیدی صورت نمی‌گیرد، این عمل یک زمان سربار به حساب می‌آید.

نکته: عمل تعویض متن توسط بخشی از سیستم عامل به نام Dispatcher صورت می‌گیرد. در واقع Dispatcher مازولی است که کنترل CPU را به فرآیندی که توسط زمانبند کوتاه مدت انتخاب شده است، اعطا می‌کند.

زمانبندی فرآیندها

وظیفه زمانبند فرآیند، انتخاب یک فرآیند جهت در اختیار گرفتن پردازنده است (در واقع مشخص می‌کند که در لحظه حاضر، پردازنده در اختیار کدام فرآیند باشد).

اهداف الگوریتم‌های زمانبندی

الگوریتم‌های زمانبندی اهداف زیر را دنبال می‌کنند (در واقع موارد زیر معیارهایی جهت مقایسه عملکرد الگوریتم‌های مختلف می‌باشند):

عدالت (Fairness)

منظور از عدالت این است که فرآیندهای هم‌ارز باید از دید زمانبند، یکسان تلقی شوند. به عبارت دیگر یک فرآیند نباید از فرآیندی هم سطح خود سهم بیشتری از CPU دریافت کند. البته این امر درباره فرآیندهایی که اولویت یکسان ندارند صادق نیست و منطقی است که فرآیندی با اولویت بالاتر، سهم بیشتری از CPU دریافت کند.

زمان پاسخ (Response)

یک الگوریتم زمانبندی باید زمان پاسخ فرآیندها را به حداقل برساند. منظور از زمان پاسخ، مدت زمان بین لحظه ورود کار و لحظه شروع پاسخ می‌باشد. این معیار در سیستم‌های محاوره‌ای و تعاملی اهمیت ویژه‌ای دارد.

زمان گردش (برگشت) کار (Turnaround Time)

یک الگوریتم زمانبندی باید زمان گردش کار فرآیندها را به حداقل برساند. منظور از زمان گردش کار، مدت زمان بین لحظه ورود کار و لحظه خروج کامل آن از سیستم می‌باشد.

نکته: به تفاوت بین زمان پاسخ و زمان گردش کار دقت کنید:

زمان پاسخ، مدت زمان بین صدور فرمان و تولید اولین پاسخ آن می‌باشد اما زمان گردش کار، مدت

زمان بین ورود کار و تکمیل نهایی آن است.

زمان انتظار (Waiting Time)

یک الگوریتم زمانبندی باید زمان انتظار فرآیندها را به حداقل برساند. منظور از زمان انتظار، مجموع زمان‌هایی است که یک فرآیند در صف فرآیندهای آماده، منتظر دریافت CPU می‌باشد. واضح است که یک الگوریتم زمانبندی، بر روی مدت زمان اجرای یک فرآیند و مدت زمان ورودی - خروجی تأثیری ندارد بلکه فقط بر روی مدت زمان انتظار یک فرآیند در صف آماده مؤثر است.

بهره‌وری پردازنده (CPU Efficiency یا CPU Utilization)

یک الگوریتم زمانبندی باید بهره‌وری پردازنده را به حداکثر برساند، به این معنی که حتی الامکان در تمام زمان‌ها پردازنده مشغول باشد تا زمان‌های هدر رفته پردازنده به حداقل برسد.

توان گذردهی (توان عملیاتی) (Throughput)

یک الگوریتم زمانبندی باید توان گذردهی را به حداکثر برساند. منظور از گذردهی، تعداد فرآیندهایی است که در واحد زمان تکمیل می‌شوند.

نکته: علاوه بر معیارهای شش‌گانه ذکر شده، معیارهای زیر نیز کم و بیش برای مقایسه الگوریتم‌های زمانبندی به کار می‌روند:

- **تعادل (توازن) در استفاده از منابع:** در واقع الگوریتم زمانبندی باید باعث شود از تمام منابع سیستم به خوبی استفاده و بهره‌برداری شود. به عبارت دیگر منابع را مشغول نگه دارد.

- **پیش‌بینی پذیری:** الگوریتم زمانبندی باید به گونه‌ای عمل کند که در اجراهای مختلف و چندباره یک کار، به خصوص، زمان انتظار و زمان پاسخ تغییرات شدیدی نداشته باشند. این معیار برای کاربران اهمیت دارد.

- رعایت اصول و اولویت‌ها

- رعایت ضرب‌العجل‌ها

نکته: برخی از معیارها و اهداف فوق با هم محقق نمی‌شوند و ممکن است با یکدیگر در تضاد باشند. به عنوان مثال ممکن است جهت برقراری عدالت سیاستی اعمال شود که بهره‌وری CPU را کاهش دهد.

نکته: الگوریتم‌های زمانبندی به دو دسته تقسیم می‌شوند:

۱- Non Preemptive (غیرقابل پس‌گیری، انحصاری)

۲- Preemptive (قابل پس‌گیری، غیرانحصاری)

در زمانبندی انحصاری، هنگامی که پردازنده در اختیار فرآیندی قرار گیرد نمی‌توان پردازنده را از وی

پس گرفت، مگر این که خود داوطلبانه آن را آزاد کند و یا فرآیند خاتمه یابد اما در زمانبندی غیرانحصاری می توان پردازنده را از فرآیندی پس گرفت و به دیگری تحویل داد.

نکته: در سیستم های تعاملی، الگوریتم های غیرانحصاری (قابل پس گیری) تنها گزینه قابل قبول هستند، زیرا در غیر این صورت یک فرآیند CPU را به انحصار خود درآورده و به دیگران اجازه کار نمی دهد. شاید به نظر برسد برای سیستم های بلادرنگ تنها باید از الگوریتم های غیرانحصاری (قابل پس گیری) استفاده کرد، اما در این سیستم ها از هر دو نوع الگوریتم می توان بهره برد. از آنجا که فرآیندها در سیستم های بلادرنگ اغلب بسیار کوچک هستند و به سرعت اجرا شده و خاتمه می یابند (یا مسدود می شوند)، الگوریتم های انحصاری (غیرقابل پس گیری) نیز در این سیستم ها کاربرد دارند.

الگوریتم های زمانبندی پردازنده (CPU)

الگوریتم های متنوعی جهت زمانبندی پردازنده وجود دارند که برخی از آنها به قرار زیرند:

FCFS_

(SPT یا SPN) SJF _

RR _

(SRTN یا SRPT یا SRTF) SRT _

HRN یا HRRN _

Priority _

MLQ _

MLFQ _

LPT _

Lottery _

Guaranteed _

FSS _

نکته: برای مطرح کردن، تحلیل و بررسی مسائل مربوط به زمانبندی پردازنده از تعاریف زیر استفاده می کنیم:

۱- زمان ورود

لحظه ای که یک فرآیند به لیست فرآیندهای آماده سیستم اضافه می شود.

۲- زمان اجرا

مدت زمانی که یک فرآیند به پردازنده نیاز دارد که به آن زمان سرویس یا زمان انفجار محاسباتی

(CBT :CPU Burst Time) نیز می‌گویند.

۳- میانگین زمان انتظار

میانگین طول مدت زمانی که فرآیندها در صف قرار داشته، آماده‌اند و منتظر دریافت پردازنده هستند.

۴- میانگین زمان پاسخ

برابر میانگین زمان اجرای فرآیندها به علاوه زمان انتظار آنهاست.

نکته: در برخی از الگوریتم‌های زمانبندی، مشکلی با عنوان قحطی زدگی (گرسنگی) (Starvation) پدید می‌آید، به این ترتیب که ممکن است اجرای یک یا چند فرآیند متناوباً به تعویق بیفتد و این روند می‌تواند تا بی‌نهایت ادامه یابد یعنی ممکن است هیچ‌گاه نوبت به اجرای این فرآیندها نرسد.

الگوریتم FCFS (First Come First Served)

این الگوریتم ساده‌ترین الگوریتم زمانبندی پردازنده است. در این روش کارها با همان ترتیب ورود به سیستم، در یک صف قرار گرفته و از ابتدای صف به ترتیب، پردازنده را در اختیار می‌گیرند.

نکته: این الگوریتم، FIFO (First In First Out) نیز نامیده می‌شود.

نکته: FCFS یک الگوریتم انحصاری (Non Preemptive) است.

نکته: الگوریتم FIFO مشکل قحطی زدگی ندارد.

مثال: چهار فرآیند p_1, p_2, p_3, p_4 را در نظر بگیرید که به ترتیب از p_1 تا p_4 وارد شده‌اند (همگی در لحظه ۰ و با اختلاف اندک وارد شده‌اند). با توجه به جدول زیر، میانگین زمان انتظار برای این چهار فرآیند چقدر است؟

فرآیند	مدت زمان اجرا (ms)
p_1	۴
p_2	۳
p_3	۲
p_4	۴

حل:

فرآیندها به ترتیب p_1, p_2, p_3, p_4 وارد شده‌اند، بنابراین داریم:

۰	۴	۷	۹	۱۳
P_1	P_2	P_3	P_4	

به نمودار فوق، نمودار گانت (Gantt Chart) گویند.

با توجه به نمودار فوق، فرآیند p_1 به محض ورود، پردازنده را در اختیار می‌گیرد، بنابراین زمان انتظار آن

برابر صفر است. فرآیند p_2 ، ۴ میلی ثانیه، فرآیند p_3 ، ۷ میلی ثانیه و فرآیند p_4 ، ۹ میلی ثانیه منتظر می‌مانند. به این ترتیب میانگین زمان انتظار فرآیندها برابر است با:

$$\frac{0 + 4 + 7 + 9}{4} = 5 \text{ میلی ثانیه}$$

دقت کنید که اگر ترتیب ورود فرآیندها به صورت p_3 ، p_2 ، p_1 و p_4 بود مسئله به این صورت تبدیل می‌شد:

۰	۲	۵	۹	۱۳
P_3	P_2	P_1	P_4	

و میانگین زمان انتظار برابر بود با:

$$\frac{0 + 2 + 5 + 9}{4} = 4 \text{ میلی ثانیه}$$

الگوریتم SJF (Shortest Job First)

در این روش ابتدا کاری برای اجرا انتخاب می‌شود که از همه کوتاهتر باشد (زمان اجرای کمتری داشته باشد).

نکته: این الگوریتم، SPN (Shortest Process Next) و

SPT (Shortest Processing Time) نیز نامیده می‌شود.

نکته: SJF یک الگوریتم انحصاری (Non Preemptive) است.

نکته: یک نقص عمده الگوریتم SJF این است که ممکن است باعث قحطی زدگی فرآیندهای طولانی شود. به این ترتیب که اگر همواره تعدادی فرآیند کوچک وارد سیستم شوند، اجرای فرآیندهای بزرگ به طور متناوب به تعویق می‌افتد. این روال حتی می‌تواند تا بی‌نهایت ادامه یابد و هیچگاه نوبت به اجرای فرآیندهای بزرگ نرسد!!!

نکته: در این روش اگر دو فرآیند مدت زمان اجرای برابر داشته باشند، براساس FCFS زمانبندی می‌شوند.

مثال: چهار فرآیند زیر را در نظر بگیرید.

فرآیند	CBT	زمان ورود
p_1	۱۵	۰
p_2	۵	۰
p_3	۸	۰
p_4	۳	۰

میانگین زمان انتظار و میانگین زمان پاسخ را برای این چهار فرآیند و با دو روش SJF و FCFS به دست

آورید (برای الگوریتم FCFS فرض کنید فرآیندها به ترتیب نامشان وارد می‌شوند).

حل: نمودار گانت برای الگوریتم FCFS به صورت زیر است:

۰	۱۵	۲۰	۲۸	۳۱
P_1	P_2	P_3	P_4	

بنابراین زمان انتظار فرآیندها در FCFS برابرند با:

$$p_1 = \text{زمان انتظار} = ۰$$

$$p_2 = \text{زمان انتظار} = ۱۵$$

$$p_3 = \text{زمان انتظار} = ۲۰$$

$$p_4 = \text{زمان انتظار} = ۲۸$$

و میانگین زمان انتظار در FCFS برابر است با:

$$\frac{۰ + ۱۵ + ۲۰ + ۲۸}{۴} = ۱۵/۷۵$$

زمان پاسخ فرآیندها در FCFS برابرند با:

$$p_1 = \text{زمان پاسخ} = ۱۵$$

$$p_2 = \text{زمان پاسخ} = ۲۰$$

$$p_3 = \text{زمان پاسخ} = ۲۸$$

$$p_4 = \text{زمان پاسخ} = ۳۱$$

و میانگین زمان پاسخ در FCFS برابر است با:

$$\frac{۱۵ + ۲۰ + ۲۸ + ۳۱}{۴} = ۲۳/۵$$

همچنین برای الگوریتم SJF داریم:

۰	۳	۸	۱۶	۳۱
P_4	P_2	P_3	P_1	

بنابراین زمان انتظار فرآیندها در SJF برابرند با:

$$p_1 = \text{زمان انتظار} = ۱۶$$

$$p_2 = \text{زمان انتظار} = ۳$$

$$p_3 = \text{زمان انتظار} = ۸$$

$$p_4 = \text{زمان انتظار} = ۰$$

و میانگین زمان انتظار در SJF برابر است با:

$$\frac{۱۶ + ۳ + ۸ + ۰}{۴} = ۶/۷۵$$

زمان پاسخ فرآیندها در SJF برابر است با:

$$p_1 = \text{زمان پاسخ} = 31$$

$$p_2 = \text{زمان پاسخ} = 8$$

$$p_3 = \text{زمان پاسخ} = 16$$

$$p_4 = \text{زمان پاسخ} = 3$$

و میانگین زمان پاسخ در SJF برابر است با:

$$\frac{31 + 8 + 16 + 3}{4} = 14.5$$

نکته: هدف الگوریتم SJF به حداقل رساندن میانگین زمان انتظار، میانگین زمان پاسخ و میانگین زمان گردش کار فرآیندهاست.

نکته: در عمل نمی‌توان الگوریتم SJF را پیاده سازی کرد، زیرا سیستم عامل زمان اجرای فرآیندها را از قبل نمی‌داند و تنها کاری که می‌تواند انجام دهد این است که زمان اجرای فرآیندها را فقط حدس زده و به طور تقریبی به دست آورد.

الگوریتم RR (Round Robin)

الگوریتم RR (نوبت چرخشی) یکی از پرکاربردترین الگوریتم‌ها در سیستم‌های اشتراک زمانی است. درباره‌ی نحوه‌ی عملکرد آن می‌توان گفت این الگوریتم نسخه‌ی غیر انحصاری (Preemptive) الگوریتم FCFS می‌باشد. در این الگوریتم زمان پردازنده را به برش‌های زمانی کوتاهی (Time Slice) تقسیم می‌کنیم. همانند الگوریتم FCFS، فرآیندهایی که به سیستم تحویل داده می‌شوند به انتهای یک صف وارد می‌شوند. سپس پردازنده از ابتدای صف شروع کرده و به هر فرآیند حداکثر به اندازه‌ی یک برش زمانی سرویس می‌دهد. در واقع پس از اینکه برش زمانی یک فرآیند به پایان رسید، پردازنده آن فرآیند را رها کرده و به سراغ فرآیند بعدی موجود در صف می‌رود. این عمل آنقدر تکرار می‌شود تا پردازنده به انتهای صف فرآیندهای آماده برسد.

به عبارت دیگر فرآیندها در یک صف دایره‌ای شکل سازماندهی می‌شوند و پردازنده به صورت دوار در این صف حرکت کرده و به هر فرآیند فقط به اندازه‌ی حداکثر یک برش زمانی سرویس می‌دهد.

نکته: به برش زمانی، کوانتوم زمانی (Quantum Time) نیز می‌گویند.

مثال: سه فرآیند p_1 ، p_2 و p_3 را در نظر بگیرید. با استفاده از الگوریتم RR و با برش زمانی ۱ ms، میانگین زمان انتظار و میانگین زمان پاسخ فرآیندها را به دست آورید (فرض کنید فرآیندها همه در لحظه‌ی صفر و به ترتیب نام وارد شده‌اند).

فرآیند	CBT	زمان ورود
p _۱	۲	۰
p _۲	۳	۰
p _۳	۲	۰

حل: با استفاده از الگوریتم RR و با برش زمانی ۱ ms، نمودار گانت به صورت زیر است:

۰	۱	۲	۳	۴	۵	۶	۷
P _۱	P _۲	P _۲	P _۱	P _۲	P _۲	P _۲	P _۲

بنابراین داریم:

فرآیند	زمان پاسخ	زمان انتظار
p _۱	۴	۲
p _۲	۷	۴
p _۳	۶	۴

$$\text{میانگین زمان انتظار} = \frac{۲ + ۴ + ۴}{۳} = ۳/۳$$

$$\text{میانگین زمان پاسخ} = \frac{۴ + ۷ + ۶}{۳} = ۵/۶$$

نکته: الگوریتم RR مشکل قحطی زدگی ندارد.

نکته: اگر حین اجرای الگوریتم RR، یک فرآیند تازه وارد و یک فرآیند قدیمی، هر دو به انتهای صف آماده برسند، فرآیندی که تازه وارد شده، جلوتر و فرآیند قدیمی، در انتها قرار می‌گیرد. البته این مسأله در همه سیستم‌ها رعایت نمی‌شود.

نکته: یکی از اهدافی که الگوریتم RR کمابیش به آن نزدیک می‌شود، رعایت عدالت و مساوات است. **نکته:** الگوریتم RR در سیستم‌های اشتراکی زمانی به خوبی استفاده می‌شود. در واقع با این الگوریتم کاربران تصور نمی‌کنند که همه فرآیندها به موازات هم در حال اجرا هستند!

نکته: برای به دست آوردن زمان پاسخ یک فرآیند، کافی است فاصله بین زمان خروج فرآیند و زمان ورود آن را به دست آورد و برای به دست آوردن زمان انتظار کافی است زمان اجرای یک فرآیند را از زمان پاسخ آن کم کرد.

زمان ورود - زمان خروج = زمان پاسخ

زمان اجرا - زمان پاسخ = زمان انتظار

نکته: اگر کار فرآیندی پیش از پایان برش زمانی، به پایان برسد، داوطلبانه پردازنده را در نیمه‌ی برش

زمانی رها می‌کند و پردازنده در اختیار فرآیند بعدی قرار می‌گیرد.

مثال: سه فرآیند زیر را در نظر بگیرید. با استفاده از الگوریتم RR و با برش زمانی ۵ میلی‌ثانیه، میانگین زمان پاسخ و میانگین زمان انتظار را محاسبه کنید.

فرآیند	CBT	زمان ورود
p_1	۱۷	۰
p_2	۳	۱
p_3	۷	۲

حل:

۰	۵	۸	۱۳	۱۸	۲۰	۲۵	۲۷
P_1	P_2	P_3	P_1	P_2	P_1	P_3	

و برای محاسبه‌ی میانگین زمان پاسخ و میانگین زمان انتظار داریم:

فرآیند	زمان پاسخ	زمان انتظار
p_1	$۲۷ - ۰ = ۲۷$	$۲۷ - ۱۷ = ۱۰$
p_2	$۸ - ۱ = ۷$	$۷ - ۳ = ۴$
p_3	$۲۰ - ۲ = ۱۸$	$۱۸ - ۷ = ۱۱$

$$\text{میانگین زمان پاسخ} = \frac{۲۷ + ۷ + ۱۸}{۳} = ۱۷/۳$$

$$\text{میانگین زمان انتظار} = \frac{۱۰ + ۴ + ۱۱}{۳} = ۸/۳$$

نکته: فرآیند تعویض متن (Context Switch) یک زمان تلف شده برای سیستم می‌باشد. به همین دلیل اندازه‌ی آن باید نسبت به طول برش زمانی بسیار کوچکتر باشد تا کارایی سیستم کاهش نیابد (در بیشتر مسائل زمان تعویض متن را نادیده می‌گیرند).

نکته: در الگوریتم RR، اگر برش زمانی بسیار بزرگ در نظر گرفته شود، این الگوریتم به FCFS تبدیل می‌شود.

نکته: افزایش اندازه‌ی برش زمانی بر روی میانگین زمان پاسخ فرآیندها تأثیر می‌گذارد. اما این تأثیر گاهی کاهش میانگین زمان پاسخ است و گاهی افزایش آن! بنابراین لزوماً نمی‌توان گفت افزایش اندازه‌ی برش زمانی، میانگین زمان پاسخ فرآیندها را افزایش یا کاهش می‌دهد، بلکه باید در هر مورد محاسبه

شود.

نکته: حد پایین یک برش زمانی توسط دو عامل مشخص می‌شود:

(۱) طول مدت زمان تعویض متن، یک برش زمانی نباید آنقدر کوچک باشد که هزینه‌های تعویض متن بر کارایی سیستم غلبه کند.

(۲) برش زمانی باید کمی بزرگتر از زمان لازم برای یک فعل و انفعال نوعی باشد، زیرا در غیر این صورت هر کاری احتیاج به حداقل دو برش زمانی خواهد داشت.

نکته: اگر اندازه‌ی برش زمانی، نسبت به مدت زمان اجرای فرآیندها بسیار کوچک باشد، این گونه به نظر می‌رسد که زمان CPU بین همه‌ی فرآیندها به طور مساوی به اشتراک گذاشته شده است و هر فرآیند کسری از CPU را در اختیار دارد.

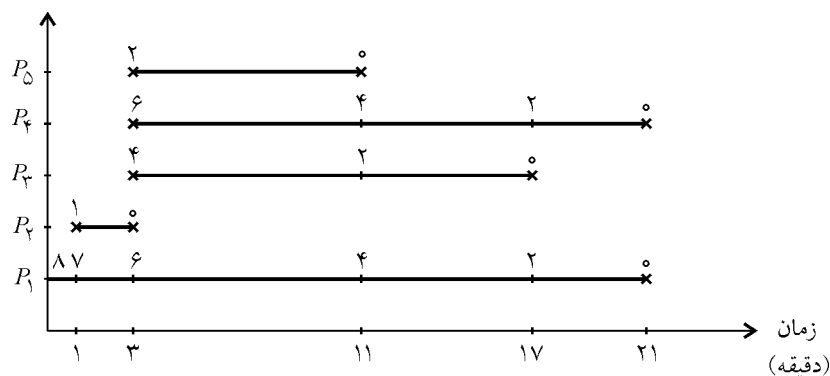
مثال: فرض کنید پنج فرآیند p_1 تا p_5 در یک سیستم وجود داشته باشند. اگر در این سیستم از روش RR با برش زمانی ۱ میلی ثانیه استفاده شود. میانگین زمان پاسخ چقدر است؟

فرآیند \	p_1	p_2	p_3	p_4	p_5
CBT به دقیقه	۸	۱	۴	۶	۲
زمان ورود به دقیقه	۰	۱	۳	۳	۳

حل: همان طور که مشاهده می‌شود زمان‌های CBT در مقیاس دقیقه هستند و برش زمانی برابر با ۱ میلی ثانیه است.

در این روش اگر n فرآیند در سیستم موجود باشند، مانند این است که هر کدام یک پردازنده با سرعت $\frac{1}{n}$ پردازنده‌ی سیستم برای خود دارند.

برای این مسأله نمودار ابداعی زیر را تحلیل می‌کنیم (اعدادی که بالای نمودار هر فرآیند درج شده، زمان باقیمانده‌ی هر فرآیند می‌باشد):



در لحظه‌ی صفر تنها فرآیند p_1 در سیستم وجود دارد، پس پردازنده فقط به آن سرویس می‌دهد و تا دقیقه ۱، یک دقیقه از کار p_1 انجام شده و در دقیقه ۱ فرآیند p_2 نیز وارد می‌شود. بنابراین از این لحظه به بعد CPU بین این دو به اشتراک گذاشته می‌شود. پس تا دقیقه ۳، و طی دو دقیقه، از هر یک از فرآیندهای p_1 و p_2 ، یک دقیقه کم می‌شود. در دقیقه ۳، فرآیند p_2 خاتمه می‌یابد اما سه فرآیند جدید به سیستم اضافه می‌شوند و از این لحظه، CPU باید بین چهار فرآیند به اشتراک گذاشته شود. پس طی هشت دقیقه (از ۳ تا ۱۱) به هر فرآیند جمعاً دو دقیقه سرویس داده می‌شود و الی آخر. بنابراین زمان پاسخ فرآیندها برابر است با:

فرآیند	زمان پاسخ (دقیقه)
p_1	$21 - 0 = 21$
p_2	$3 - 1 = 2$
p_3	$17 - 3 = 14$
p_4	$21 - 3 = 18$
p_5	$11 - 3 = 8$

$$\text{میانگین زمان پاسخ} = \frac{21 + 2 + 14 + 18 + 8}{5} = 12/6$$

الگوریتم SRT (Shortest Remaining Time)

این الگوریتم نسخه غیرانحصاری (Preemptive) الگوریتم SJF است. در این الگوریتم اگر حین اجرای یک فرآیند، فرآیندی وارد شود که زمان اجرای کوتاه‌تری داشته باشد، پردازنده را در اختیار می‌گیرد.

نکته: این الگوریتم، SRPT (Shortest Remaining Processing Time)،

و SRTF (Shortest Remaining Time First)

SRTN (Shortest Remaining Time Next) نیز نامیده می‌شود.

نکته: اگر لحظه ورود همه فرآیندها یکی باشد، الگوریتم SRT مشابه SJF عمل می‌کند.

نکته: در الگوریتم SRT نیز همانند الگوریتم SJF، احتمال وقوع قحطی‌زدگی برای کارهای بزرگ وجود دارد.

مثال: چهار فرآیند p_1 تا p_4 را در نظر بگیرید. با روش SRT، میانگین زمان انتظار برای فرآیندها چقدر است؟

فرآیند	p_1	p_2	p_3	p_4
CBT	۸	۴	۹	۵
زمان ورود	۰	۱	۲	۳

حل: با استفاده از روش SRT، نمودار گانت به صورت زیر است:

۰	۱	۵	۱۰	۱۷	۲۶
P_1	P_2	P_4	P_3	P_1	P_2

باید دقت کرد در لحظه ۱ که p_2 وارد می شود چون زمان باقیمانده آن از زمان باقیمانده p_1 کمتر است، پردازنده از p_1 گرفته شده و به p_2 داده می شود.

$$\text{میانگین زمان انتظار} = \frac{9 + 0 + 15 + 2}{4} = \frac{26}{4} = 6.5$$

الگوریتم HRRN (Highest Response Ratio Next)

الگوریتم های SJF و SRT مشکل قحطی زدگی دارند. در این الگوریتم ها به فرآیندهای کوچک بیش از اندازه توجه می شود! برای رفع این مشکل الگوریتم HRRN مورد استفاده قرار می گیرد. در این الگوریتم اولویت فرآیندها برای اجرا، فقط اندازه آن ها نیست. برای تعیین اولویت یک فرآیند در الگوریتم HRRN از فرمول زیر استفاده می شود:

$$\text{اولویت} = \frac{\text{زمان انتظار} + \text{زمان اجرا}}{\text{زمان اجرا}}$$

در این فرمول، از آن جا که زمان اجرا در مخرج کسر قرار دارد، در نتیجه فرآیندهای کوچکتر اولویت بالاتری دارند اما از آن جا که زمان انتظار در صورت کسر قرار دارد، هر چه یک فرآیند بیشتر منتظر دریافت پردازنده بماند، اولویت بالاتری به دست می آورد. با این روش هم زمان اجرای یک فرآیند در تعیین اولویت آن تأثیر دارد و هم مدت زمانی که منتظر می ماند.

نکته: این الگوریتم HRRN نیز نامیده می شود.

نکته: الگوریتم HRRN مشکل قحطی زدگی ندارد.

نکته: HRRN یک الگوریتم انحصاری (Non Preemptive) است.

مثال: میانگین زمان پاسخ و میانگین زمان انتظار را برای چهار فرآیند زیر به روش HRRN محاسبه کنید.

فرآیند	زمان ورود	زمان اجرا
p_1	۰	۱۱
p_2	۶	۵
p_3	۸	۴
p_4	۱۰	۲

حل: در لحظه صفر، فقط p_1 وارد می شود، بنابراین به صورت انحصاری پردازنده را در اختیار می گیرد و زمانی که اجرای آن به پایان رسید، برای تمام فرآیندهای موجود، اولویت محاسبه می شود و فرآیندی انتخاب می شود که از همه اولویت بالاتری داشته باشد. این روال به همین شکل تا انتها ادامه می یابد.

۰	۱۱	۱۶	۱۸	۲۲
P_1	P_2	P_4	P_3	

در لحظه ۱۱ اولویت ها به صورت زیر هستند:

$$pp_2 = \frac{5+5}{5} = 2$$

$$pp_3 = \frac{3+4}{4} = 1/75$$

$$pp_4 = \frac{1+2}{2} = 1/5$$

بنابراین در لحظه ۱۱ فرآیند p_2 انتخاب می شود.

در لحظه ۱۶ اولویت ها به صورت زیر هستند:

$$pp_3 = \frac{8+4}{4} = 3$$

$$pp_4 = \frac{6+2}{2} = 4$$

بنابراین در لحظه ۱۶ فرآیند p_4 انتخاب می شود و در لحظه ۱۸ نیز فقط فرآیند p_3 موجود می باشد.

در نهایت داریم:

فرآیند	زمان انتظار	زمان پاسخ
p_1	۰	۱۱
p_2	$11-6=5$	۱۰
p_3	$18-8=10$	۱۴
p_4	$16-10=6$	۱۸

$$\text{میانگین زمان انتظار} = \frac{0+5+10+6}{4} = 5/25$$

$$\text{میانگین زمان پاسخ} = \frac{11+10+14+18}{4} = 10/75$$

الگوریتم های Priority (زمانبندی با اولویت)

در این روش زمانبندی، هر یک از فرآیندها اولویت مخصوص به خود را دارند. این اولویت معمولاً از خارج سیستم مشخص می شود. منطقی به نظر می رسد که اولویت فرآیندی مربوط به رئیس از اولویت فرآیندی مربوط به کارمند بالاتر باشد. ایده اصلی این الگوریتم بسیار ساده و مشخص است. هر فرآیند

باید یک اولویت داشته باشد و در هر لحظه فرآیندی اجرا می‌شود که بالاترین اولویت را دارد.
نکته: الگوریتم‌های Priority را می‌توان هم به صورت انحصاری و هم به صورت غیرانحصاری پیاده‌سازی کرد.

نکته: تنوع الگوریتم‌های زمانبندی با اولویت بسیار زیاد است و انواع مختلفی از آن وجود دارد که به عنوان مثال می‌توان به SRT، SJF و HRRN اشاره کرد. البته در این سه الگوریتم، اولویت فرآیندها در داخل سیستم و براساس شرایط مشخص می‌شود.

نکته: در الگوریتم‌های Priority، فرآیندهای با اولویت کمتر، دچار قحطی زدگی می‌شوند.

نکته: یک اولویت می‌تواند استاتیک یا دینامیک باشد:

- یک اولویت استاتیک هیچ‌گاه تغییر نمی‌کند، به همین دلیل پیاده‌سازی آن ساده است.

- یک اولویت دینامیک بر اثر تغییراتی که در محیط اتفاق می‌افتد تغییر می‌کند.

نکته: در الگوریتم‌های اولویت، جهت مقابله با مشکل قحطی زدگی برخی از فرآیندها، می‌توان از تکنیکی موسوم به سالخوردگی (Aging) استفاده کرد. در این تکنیک به تدریج اولویت پردازش‌هایی که مدت مدیدی در انتظار بوده‌اند، افزایش می‌یابد.

الگوریتم‌های MLQ (Multi Level Queues)

در این روش که به آن صفحات چندسطحی گویند، فرآیندها را به چند دسته تقسیم کرده و فرآیندهای موجود در هر دسته را در یک صف قرار می‌دهیم. در این حالت هر صف اولویت خاص خود را دارد و صف‌ها به ترتیب اولویت چیده می‌شوند. هر صف در داخل خود می‌تواند از الگوریتم زمانبندی جداگانه‌ای استفاده کند.

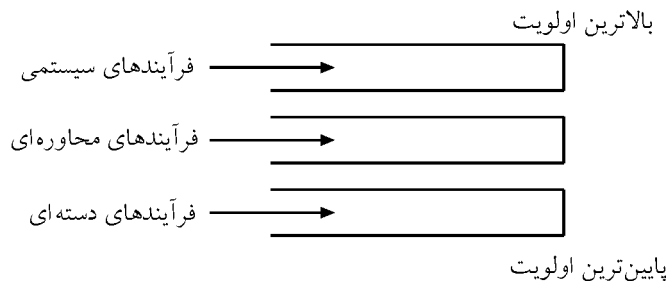
به عنوان مثال فرض کنید همه فرآیندهای سیستم در یکی از گونه‌های زیر قرار گیرند:

۱- فرآیندهای سیستمی

۲- فرآیندهای محاوره‌ای

۳- فرآیندهای دسته‌ای

در این حالت سیستم عامل برای هر گروه یک صف جداگانه تشکیل می‌دهد و فرآیندهای جدید را با توجه به نوعشان به صف موردنظر هدایت می‌کند. هر یک از صف‌ها اولویت خاص خود را دارد. برای مثال در این سیستم، صف فرآیندهای سیستمی از بالاترین اولویت برخوردار است.



مثالی از الگوریتم MLQ

در روش MLQ، هر صف الگوریتم زمانبندی خاص خود را دارد. برای مثال در صف فرآیندهای سیستمی می‌توان از روش SJF، در صف فرآیندهای محاوره‌ای از روش RR و در صف فرآیندهای دسته‌ای از روش FCFS استفاده کرد. نکته مهم در این الگوریتم این است که نحوه تخصیص پردازنده بین صف‌ها چگونه باشد؟ برای این منظور دو رویکرد وجود دارد، انحصاری و غیرانحصاری.

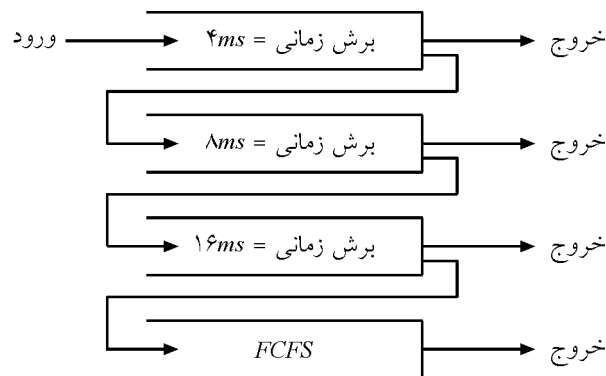
در رویکرد انحصاری، پردازنده به صورت غیرقابل پس‌گیری به صف‌ها داده می‌شود، بنابراین ابتدا به صف با بالاترین اولویت اختصاص می‌یابد و براساس الگوریتم مربوط به آن صف، به فرآیندهای آن سرویس می‌دهد. در این روش تا فرآیندهای موجود در یک صف با اولویت بالاتر به اتمام نرسیده باشند، پردازنده به صف با اولویت پایین‌تر تخصیص نمی‌یابد. از طرفی حتی اگر پردازنده مشغول سرویس دادن به صف سوم باشد و در این لحظه، یک فرآیند با اولویت بالاتر مثلاً برای صف اول یا دوم وارد شود، پردازنده صف سوم را رها کرده و به سراغ صف بالاتر می‌رود. واضح است که در این حالت احتمال وقوع قحطی‌زدگی وجود دارد.

اما در رویکرد غیرانحصاری، پردازنده با استفاده از روشی مشابه RR، بین صف‌ها حرکت می‌کند. مثلاً می‌توان ۵۰٪ زمان پردازنده را به صف اول، ۳۰٪ را به صف دوم و در نهایت ۲۰٪ را به صف سوم اختصاص داد.

الگوریتم MLFQ (Multi Level Feedback Queues)

روش MLFQ در واقع تغییر یافته الگوریتم MLQ است. در الگوریتم MLQ یک فرآیند پس از ورود به یک صف خاص، تا پایان در همان صف باقی می‌ماند، اما ایده موجود در روش MLFQ این است که فرآیندها با توجه به رفتارشان بین صف‌ها حرکت کنند. به عنوان مثال فرآیندی که در یک صف با اولویت بالا قرار دارد و زمان پردازنده را زیاد مصرف می‌کند، به صف پایین‌تر منتقل می‌شود و یا فرآیندی که در یک صف با اولویت پایین بیش از حد منتظر دریافت پردازنده مانده است، به صف بالاتر منتقل می‌شود.

مثال: فرض کنید سیستمی با ۴ صف در اختیار داریم، در صف اول از روش RR با برش زمانی ۴ms، در صف دوم از روش RR با برش زمانی ۸ms، در صف سوم از روش RR با برش زمانی ۱۶ms و در صف آخر از روش FCFS استفاده می‌کنیم (شکل زیر). در این حالت فرآیندها ابتدا به صف اول وارد می‌شوند، اگر در این صف به پایان نرسیدند به انتهای صف دوم، اگر در این صف نیز کامل نشدند به انتهای صف سوم و اگر باز هم به پایان نرسیدند به انتهای صف چهارم وارد می‌شوند که از روش FCFS استفاده می‌کنند.



مثالی از الگوریتم MLFQ

الگوریتم LPT (Longest Processing Time)

این روش که معمولاً به صورت **انحصاری** پیاده‌سازی می‌شود، برعکس روش SJF می‌باشد، به این معنا که پردازنده از بین کارهای موجود، طولانی‌ترین کار را انتخاب می‌کند.

نکته: در الگوریتم LPT مشکل **قحطی‌زدگی** برای کارهای کوچک وجود دارد.

نکته: این الگوریتم میانگین زمان پاسخ و انتظار را حداکثر می‌کند!!!

الگوریتم Lottery

ایده موجود در این روش از ایده بخت‌آزمایی گرفته شده است. در این روش تعدادی بلیط بین فرآیندها تقسیم شده، سپس در ابتدای هر برش زمانی، یکی از بلیط‌ها به طور تصادفی برنده اعلام می‌شود و فرآیندی که آن بلیط را در اختیار دارد، پردازنده را برای آن برش زمانی در اختیار می‌گیرد.

در این روش فرآیندهای با اولویت بالاتر، بلیط‌های بیشتری در اختیار دارند، بنابراین شانس بیشتری برای در اختیار گرفتن پردازنده خواهند داشت.

نکته: الگوریتم Lottery مشکل **قحطی‌زدگی** ندارد.

نکته: فرآیندها می‌توانند بلیط‌های خود را با هم مبادله کنند. برای مثال فرآیندی که منتظر I/O است می‌تواند بلیط‌های خود را به فرآیندهای دیگر دهد.

نکته: دلیل **غیرانحصاری** بودن الگوریتم Lottery آن است که هنگامی که یک فرآیند پردازنده را در

اختیار دارد، پس از پایان برش زمانی پردازنده از وی گرفته شده و به فرآیند دیگری که بلیط اعلام شده بعدی را در اختیار دارد داده می شود.

الگوریتم Guaranteed (زمان بندی تضمین شده)

یکی از روش های متفاوت در زمان بندی، Guaranteed Scheduling است. در این روش ابتدا با کاربران درباره سهمشان از پردازنده توافق شده، سپس سهم هر یک داده می شود. یک مثال بسیار ساده این است که در یک سیستم چند کاربره با n فرآیند، اگر همه چیز را یکسان فرض کنیم، به هر فرآیند قول $\frac{1}{n}$ زمان پردازنده را می دهیم و هر فرآیند باید $\frac{1}{n}$ از زمان پردازنده را دریافت کند. برای نیل به این هدف سیستم عامل باید برای هر فرآیند یک حساب باز کند! به این معنا که باید دقیقاً بداند هر فرآیند از ابتدا تا کنون چه مدت از پردازنده استفاده کرده و همچنین چه مقدار از سهم توافق شده باقی مانده است.

الگوریتم FSS (Fair Share Scheduling) (زمان بندی سهم عادلانه)

در الگوریتم FSS، قبل از زمان بندی فرآیندها این نکته باید در نظر گرفته شود که هر فرآیند متعلق به چه کسی است، در واقع در این حالت سهم هر کاربر مشخص می شود و زمان بند به گونه ای فرآیندها را انتخاب می کند که این سهم رعایت شود.

به عنوان مثال فرض کنید در یک سیستم دو کاربر A و B وجود دارند که کاربر A جمعاً ۹ فرآیند و کاربر B فقط ۱ فرآیند در حال اجرا دارد. در این حالت اگر مثلاً از الگوریتم RR استفاده کنیم، کاربر A جمعاً ۹۰٪ زمان پردازنده را از آن خود کرده است و به کاربر B فقط ۱۰٪ زمان پردازنده می رسد! جهت جلوگیری از این وضعیت، الگوریتم FSS این نکته را در نظر می گیرد که فرآیندها متعلق به چه کسی هستند. مثلاً در سیستم قبل که دو کاربر A و B را داشتیم اگر به هر کدام ۵۰٪ از پردازنده را وعده داده باشیم، هر کدام از کاربرها بدون توجه به این که چه تعداد فرآیند دارند، سهم خود را دریافت می کنند.

برای مثال فرض کنید کاربر A ، چهار فرآیند A_1, A_2, A_3, A_4 و کاربر B فقط یک فرآیند B را در اختیار دارد. اگر سهم هر کاربر از پردازنده ۵۰٪ باشد، با زمان بندی FSS روال تخصیص پردازنده به فرآیندها می تواند به صورت زیر باشد:

$A_1 BA_2 BA_3 BA_4 BA_1 BA_2 BA_3 BA_4 B...$

اما مثلاً اگر سهم کاربر A ، ۷۵٪ و سهم کاربر B ، ۲۵٪ باشد، دنباله زیر می تواند حاصل شود.

$A_1 A_2 A_3 BA_4 A_1 A_2 BA_3 A_4 A_1 BA_2 A_3 A_4 B...$

کنکور کارشناسی ارشد

سیستم عامل

مؤلف:

ارسطو خلیلی فر



انتشارات آزاده

تقدیم به:

تمامی آنانی که برای پیشرفت و سعادت خود و بشریت
تلاش می کنند.

ارسطو خلیلی فر

به نام خدا

مقدمه مولف

کترین برتر اندیشه برگزیده

به نام خداوند جان و خرد

کتاب حاضر، کامل‌ترین مرجع حل تشریحی سوالات درس سیستم عامل، ویژه کنکور کارشناسی ارشد مهندسی کامپیوتر و مهندسی فناوری اطلاعات می‌باشد. با توجه به اهمیت خاصی که درس سیستم عامل برای موفقیت در آزمون کارشناسی ارشد دارد، بعد از سال‌ها تدریس و تحقیق، تصمیم گرفتیم، این مجموعه خاص را به شیوه‌ای منحصربه‌فرد جمع‌آوری نمایم. در حال حاضر اکثر اساتید دانشگاه‌های معتبر کشور از کتاب‌های سه نویسنده مشهور این درس یعنی آبراهام سیلبرشاتز، ویلیام استالینگز و اندرو اس تن‌ن‌بام استفاده می‌کنند و عموم تست‌های مطرح شده در چند سال اخیر از مفاهیم همین کتاب‌ها بوده است. لذا بر آن شدم تا با بهره‌گیری از نوشته‌های این کتب مرجع و برخی منابع معتبر دیگر، کتابی را به صورت جامع به رشته تحریر درآورم. این کتاب تلاش نموده است تا مباحث مطرح شده را بر مبنای تدریس دانشگاهی و براساس سرفصل‌های مصوب وزارت علوم، تحقیقات و فناوری و منطبق با مفاهیم و مباحث تدریس شده در دانشگاه‌های معتبر ایران و جهان بررسی و تبیین نماید.

این کتاب در هشت فصل و براساس جدیدترین تغییرات منابع و آزمون‌های کارشناسی ارشد تنظیم شده است.

این کتاب علاوه بر اینکه برای داوطلبان آزمون کارشناسی ارشد، قابل استفاده است، مرجع ارزنده و جامعی برای آموزش این درس در دانشگاه‌ها می‌باشد.

با کمال میل خرسندم که مراتب سپاسگزاری و قدردانی خود را از همه عزیزانی که در تهیه این کتاب نقش داشته‌اند به واسطه حمایت‌های بی‌دریغشان تقدیم نمایم.

از جناب آقای جعفر بدوستانی ریاست محترم انتشارات آزاده (راهیان ارشد) که در به ثمر رساندن این اثر بسیار محبت کرده‌اند، سپاسگزارم.

جناب آقای امیر بدوستانی مدیریت محترم بخش نشر و چاپ انتشارات به پاس سخت‌کوشی و تلاش فراوانشان، کمال تشکر و قدردانی را می‌نمایم.

هم‌چنین سرکار خانم فرزانه محمدلو که انجام امور اجرایی کتاب را برعهده داشتند، کمال تشکر و قدردانی را می‌نمایم.

در پایان دوست دارم در یک بیان صمیمانه، به خوانندگان محترم، ابراز کنم که در نوشتن این کتاب چیزی به جز عشق و خدمت را لحاظ نکرده‌ام. تمام فکرم آن بوده که هرآنچه در توان دارم را در این مسیر به کار گیرم.

با وجود دقت فراوانی که در تهیه این اثر به کار رفته است، وجود اشتباه در آن اجتناب‌ناپذیر است. لذا از تمامی دوستان، اساتید و دانشجویان عزیز خواهشمندم هرگونه نظر و پیشنهاد در زمینه اصلاح یا بهبود این کتاب را از طریق سایت:

khalilifar.ir

و یا صفحه اینستاگرام:

arastoo.khalilifar

و یا صفحه تلگرام:

[@arastookhalilifar](https://www.instagram.com/arastookhalilifar)

و یا به طور مستقیم با شماره تلفن ۰۹۱۲۲۳۰۶۶۰۲ با من در میان بگذارند.

امید است که این خدمت ناچیز مورد قبول خداوند متعال قرار گرفته و قابل استفاده شما عزیزان باشد.

ارسطو خلیلی فر

فهرست مطالب

تست‌های فصل اول: مفاهیم اولیه	۹
پاسخ تست‌های فصل اول: مفاهیم اولیه.....	۱۰
تست‌های فصل دوم: مدیریت فرآیندها و زمان‌بندی پردازنده	۱۴
پاسخ تست‌های فصل دوم: مدیریت فرآیندها و زمان‌بندی پردازنده	۱۸
تست‌های فصل سوم: مدیریت نخ.....	۴۰
پاسخ تست‌های فصل سوم: مدیریت نخ	۴۳
تست‌های فصل چهارم: مدیریت حافظه اصلی	۵۵
پاسخ تست‌های فصل چهارم: مدیریت حافظه اصلی	۶۰
تست‌های فصل پنجم: مدیریت حافظه مجازی.....	۱۱۱
پاسخ تست‌های فصل پنجم: مدیریت حافظه مجازی.....	۱۱۷
تست‌های فصل ششم: مدیریت فرآیندها و نخ‌های هم‌روند	۱۵۱
پاسخ تست‌های فصل ششم: مدیریت فرآیندها و نخ‌های هم‌روند	۱۵۹
تست‌های فصل هفتم: مدیریت بن‌بست	۲۵۳
پاسخ تست‌های فصل هفتم: مدیریت بن‌بست	۲۵۷
تست‌های فصل هشتم: مدیریت دیسک.....	۲۷۴
پاسخ تست‌های فصل هشتم: مدیریت دیسک	۲۷۶

جهت کسب اطلاعات بیشتر درباره‌ی تازه‌ترین خبرهای آزمون
کارشناسی ارشد در گرایش‌های مختلف رشته‌ی مهندسی کامپیوتر و
مهندسی فناوری اطلاعات به سایت ارسطو خلیلی فر مراجعه نمایید:

www.khalilifar.ir

همچنین از شما درخواست می‌کنیم که سؤالات، طرح‌ها و پیشنهادات خود
را برای بهبود، تکمیل و تصحیح این کتاب با ما در میان بگذارید.
ایمیل‌های خود را به آدرس arastoo.khalilifar@gmail.com
ارسال نمایید.

انتشارات آزاده (راهیان ارشد)

تست‌های فصل دوم: مدیریت فرآیندها و زمان‌بندی پردازنده

۱- فرض کنید 5 فرآیند با مشخصات زیر به یک سیستم با زمان‌بند چرخشی (Round – Robin) با برش زمانی $q = 1$ وارد شوند: (e: execution time, r: release time)
 با فرض این که همیشه بین فرآیندی که در لحظه t برش زمانی خود را به پایان می‌رساند و فرآیند ورودی در لحظه t اولویت با فرآیند قبلی موجود در سیستم است و در شرایط کاملاً یکسان بین دو فرآیند، اولویت با فرآیند با شماره کوچک‌تر است. میانگین زمان انتظار فرآیندهای فوق کدام است؟ (مهندسی IT - دولتی، ۸۹)

$P_1 : r=0 , e=2$

$P_2 : r=0 , e=3$

$P_3 : r=1 , e=2$

$P_4 : r=1 , e=3$

$P_5 : r=2 , e=2$

5.6 (۴)

6.2 (۳)

5.8 (۲)

6.4 (۱)

۲- اگر در یک سیستم دو فرآیند P_1 و P_2 داشته باشیم که jobهای فرآیند P_1 به صورت دوره‌ای هر 5 ثانیه یکبار و jobهای فرآیند P_2 به صورت دوره‌ای هر 4 ثانیه یکبار به سیستم وارد شوند و زمان اجرای هر job از فرآیند P_1 برابر با 3 ثانیه و زمان اجرای هر job از فرآیند P_2 برابر با 1 ثانیه باشد، بهره‌وری (Utilization) و میانگین زمان پاسخ (Average Response Time) سیستم به ترتیب چه اعدادی خواهد بود؟

(الگوریتم زمان‌بندی RR با برش زمانی $q=1$ است و اگر در لحظه t یک job به سیستم وارد شود و در همین لحظه یک job دیگر پردازنده را ترک کرده و به صف آمادگی (Ready Queue) منتقل شود، اولویت با job قبلی موجود در سیستم است که تازه پردازنده را رها کرده است. زمان پاسخ، تأخیر بین ورود هر job و اولین زمان در اختیار گرفتن پردازنده توسط آن job است.)

(مهندسی کامپیوتر - دولتی، ۹۰)

$\frac{1}{3}, 90\%$ (۴)

$\frac{5}{9}, 90\%$ (۳)

$\frac{5}{9}, 90\%$ (۲)

$\frac{1}{3}, 85\%$ (۱)

۳- سه وظیفه T_1, T_2, T_3 را در نظر بگیرید که تاکنون $n+1$ بار در سیکل آماده - اجرا - مسدود طی مسیر کرده‌اند. زمان اجرای واقعی سیکل $n+1$ ام این وظایف به ترتیب 2، 4 و 6 میلی‌ثانیه و زمان برآورد شده برای اجرای $n+1$ ام آنها نیز به ترتیب 4، 6 و 6 میلی‌ثانیه می‌باشد. زمان اجرای واقعی در سیکل $n+2$ ام به ترتیب 4 و 3 و 6 است. پس از پایان سیکل $n+1$ ام وظیفه‌های T_1 و T_2 بلافاصله آماده هستند و وظیفه T_3 پس از 3 میلی‌ثانیه آماده می‌شود. با

فرض $\alpha = 0.5$ در صورتی که بخواهیم این وظایف را با استفاده از الگوریتم‌های SPN (یا SJF) و FCFS، RR زمان‌بندی کنیم، میانگین زمان کامل (Turnaround) این وظایف فقط در سیکل $n+2$ ام چقدر خواهد بود؟ زمان Context Switch را در نظر نگیرید.

(مهندسی کامپیوتر - دولتی ۹۱)

- FCFS 7, SPN 7 (۲) FCFS 7, SPN 6.7 (۱)
 SPN 7.3, RR 9 (۴) SPN 6.7, RR 9 (۳)

۴- چهار فرآیند P_1 و P_2 و P_3 و P_4 به ترتیب در زمان‌های 0 و 3 و 7 و 40 وارد می‌شوند و مشخصات آنها مطابق جدول زیر است. اطلاعات هر سطر، منبع مورد نیاز هر فرآیند و همچنین تعداد واحدهای زمانی مورد نیاز را تعیین می‌کنند. برای مثال CPU 5 بدین معنی است که فرآیند CPU را به اندازه 5 واحد نیاز دارد. این سیستم، از زمان‌بندی RR استفاده می‌کند و $q=5$ است. قبل از اینکه هر فرآیند پردازنده را در اختیار بگیرد 1 واحد زمانی سربردار دارد. میانگین زمان کل (Turnaround Time-TR) و بهره‌وری پردازنده و بهره‌وری IO چقدر است؟ (مهندسی IT - دولتی ۹۱)

P_1	P_2	P_3	P_4
CPU 5	CPU 2	CPU 8	CPU 9
IO 5	IO 22		IO 2
CPU 2	CPU 2		CPU 1

- TR = 18.75, CPU_UTIL = 95.5% IO_UTIL = 82.5% (۱)
 TR = 20.75, CPU_UTIL = 95.5% IO_UTIL = 72.5% (۲)
 TR = 18.75, CPU_UTIL = 92.5% IO_UTIL = 82.5% (۳)
 TR = 20.75, CPU_UTIL = 92.5% IO_UTIL = 72.5% (۴)

۵- اگر فرآیندهای P_0, P_1, P_2 با هم وارد سیستم شده و به ترتیب به 15، 12، 25 ثانیه زمان برای اجرا نیاز داشته باشند، در صورت استفاده از روش صف بازخورد چند سطحی (Multilevel Feedback Queue) با سه صف با کوانتوم‌های 4، 8، 16 ثانیه و زمان تعویض متن 1 ثانیه، میانگین زمان برگشت فرآیندها (Turnaround Time) چند ثانیه خواهد بود؟

(مهندسی کامپیوتر - دولتی ۹۲)

- 39 (۴) 39.66 (۳) 39.33 (۲) 40 (۱)

۶- کدام یک از انواع برنامه‌ریزها وظیفه فعال‌سازی و تعلیق فرآیندها را به عهده دارد؟

(مهندسی IT - دولتی ۹۲)

- Long-Term Scheduler (۲) Prioritized Scheduler (۱)
 Short-Term Scheduler (۴) Medium-Term Scheduler (۳)

۷- جدول زیر زمان ورود و زمان پردازش پردازش‌ها را نشان می‌دهد. برای آنکه دو روش برنامه‌ریزی First Come First Served (FCFS) و Highest Response Ratio Next (HRRN) زمان‌بندی‌های یکسانی را تولید نمایند، چه رابطه‌ای باید بین t_B و t_C برقرار باشد؟ برای سادگی از زمان تعویض زمینه بین پردازش‌ها صرف‌نظر کنید. (مهندسی کامپیوتر - دولتی ۹۳)

فرآیند	زمان ورود	زمان پردازش
A	0	$t_A = 3$
B	1	t_B
C	2	t_C

$$t_B < t_C \quad (۱) \quad t_C < t_B \quad (۲) \quad t_C < 2t_B \quad (۳) \quad t_B < 2t_C \quad (۴)$$

۸- اگر هر فرآیند در سیستم عامل بتواند در یکی از حالت‌های ایجاد، مسدود، اجرا، پایان یافته و آماده اجرا باشد، کدام یک از تغییر حالت‌های مستقیم زیر مجاز نیست؟ (مهندسی IT - دولتی ۹۳)

- (۱) آماده اجرا به اجرا
(۲) آماده اجرا به مسدود
(۳) اجرا به آماده اجرا
(۴) مسدود به آماده اجرا

۹- تحت چه شرایطی زمان‌بند Round-Robin دقیقاً مانند زمان‌بند FCFS عمل می‌نماید؟

(مهندسی IT - دولتی ۹۴)

- (۱) همه پردازش‌ها مانند هم باشند و ترتیب ورود پردازش‌ها در دو زمان‌بند یکسان باشد.
(۲) اولویت همه پردازش‌ها برابر باشد و ترتیب ورود پردازش‌ها در دو زمان‌بند یکسان باشد.
(۳) زمان اجرای همه پردازش‌ها برابر باشد و ترتیب ورود پردازش‌ها در دو زمان‌بند یکسان باشد.
(۴) زمان اجرای پردازش‌ها کوچکتر از زمان کوانتوم باشد و ترتیب ورود پردازش‌ها در دو زمان‌بند یکسان باشد.

۱۰- پنج پردازش همزمان وارد سیستم می‌شوند که زمان اجرای آنها به صورت زیر است. پردازش‌ها به ترتیب شماره آنها اجرا می‌شوند. اگر زمان‌بندی پردازش‌ها بر اساس الگوریتم Round Robin و $Quantum=2$ صورت بگیرد، زمان انتظار صف پردازش 2 و 4 چقدر است؟ (مهندسی IT - دولتی ۹۴)

P1:2

P2:6

P3:1

P4:4

P5:3

(۲) $P4=10$ و $P2=8$

(۱) $P4=10$ و $P2=10$

(۴) $P4=8$ و $P2=10$

(۳) $P4=9$ و $P2=10$

۱۱- یک سیستم تک‌پردازنده‌ای با صف بازخورد چند سطحی (Multi-level Feedback Queue) را در نظر بگیرید. به سطح اول تکه زمانی معادل ۸ میکروثانیه و به سطح دوم، تکه زمانی معادل ۱۶ میکروثانیه داده شده است و سطح سوم، به ترتیب ورود زمان‌بندی شده است (FCFS). فرض کنید ۶ کار همگی در زمان صفر وارد سطح اول سیستم شده و زمان اجرای آنها به ترتیب برابر با ۴، ۷، ۱۲، ۲۰، ۲۵ و ۳۰ میکروثانیه باشد. در این صورت میانگین زمان بازگشت (ATT: Average Turnaround Time) و میانگین زمان انتظار (AWT: Average Waiting Time) کارهای فوق در سیستم به ترتیب از راست به چپ کدام است؟ (مهندسی کامپیوتر - دولتی ۹۵)

(۱) ۵۵.۵، ۴۱.۸ (۲) ۴۵.۵، ۵۱.۸ (۳) ۳۵.۵، ۵۱.۸ (۴) ۲۵.۵، ۴۱.۸

۱۲- در جدول زیر اطلاعات مربوط به پنج پردازنده (process) که وارد سیستم می‌شوند داده شده است. اگر سیستم از روش زمان‌بندی نوبتی (RR) با تکه زمانی (کوآتوم) ۴۰ ثانیه و سربار ۱۰ ثانیه برای تعویض متن فرآیندها (context switch) استفاده کند، میانگین زمان انتظار پردازنده‌ها چقدر است؟ (مهندسی IT - دولتی ۹۵)

E	D	C	B	A	نام برنامه	48 (۱)
270	260	250	20	10	زمان ورود (s)	56 (۲)
40	30	30	50	100	مدت زمان لازم برای اجرا (s)	106 (۳)
						108 (۴)

۱۳- با توجه به جدول، متوسط زمان برگشت (Turnaround Time) و زمان انتظار (Waiting Time) پردازنده‌های زیر را به ازای الگوریتم Preemptive Shortest Remaining Job First چه عددی است؟ (مهندسی کامپیوتر - دولتی ۹۶)

پردازنده	زمان ورود به سیستم	زمان موردنیاز برای اجرا
P ₁	1	10
P ₂	3	8
P ₃	7	6
P ₄	11	3
P ₅	15	7

(۱) ۱۴ و ۶.۲ (۲) ۱۶ و ۶.۲ (۳) ۱۴ و ۷.۲ (۴) ۱۶ و ۷.۲

پاسخ تست‌های فصل دوم: مدیریت فرآیندها و زمان‌بندی پردازنده

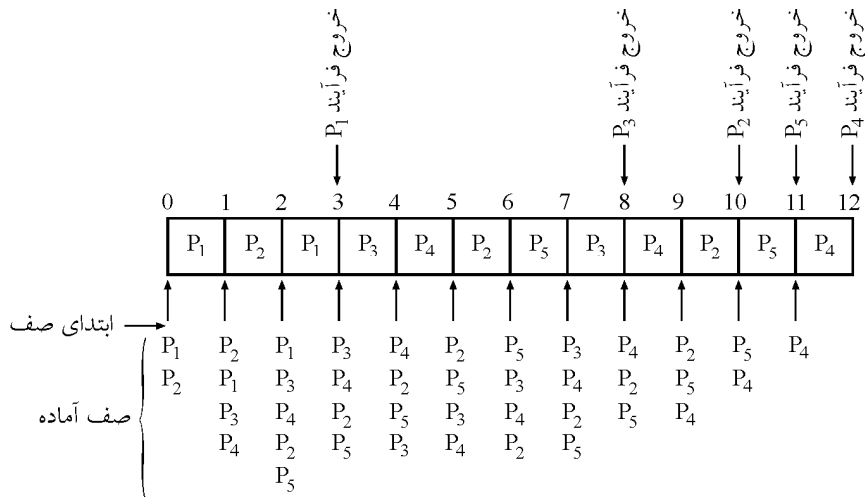
۱- گزینه (۴) صحیح است.

با توجه به مفروضات مطرح شده در صورت سؤال داریم:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار+	زمان بازگشت=
P ₁	0	2		
P ₂	0	3		
P ₃	1	2		
P ₄	1	3		
P ₅	2	2		

توجه: به فرض مطرح شده در صورت سؤال توجه فراوان داشته باشید.

فرض مطرح شده: همیشه بین فرآیندی که در لحظه t ، برش زمانی خود را به پایان می‌رساند و فرآیند ورودی در لحظه t ، اولویت با فرآیند قبلی موجود در سیستم است و در شرایط کاملاً یکسان بین دو فرآیند، اولویت با فرآیند با شماره کوچکتر است.
 با توجه به برش زمانی $q=1$ و فرض مطرح شده در صورت سؤال جهت اولویت‌بندی صف الگوریتم RR، نمودار گانت زیر را داریم:



زمان ورود فرآیند - زمان خروج فرآیند = زمان بازگشت فرآیند

P₁ زمان بازگشت = 3-0=3

$$P_2 \text{ زمان بازگشت} = 10 - 0 = 10$$

$$P_3 \text{ زمان بازگشت} = 8 - 1 = 7$$

$$P_4 \text{ زمان بازگشت} = 12 - 1 = 11$$

$$P_5 \text{ زمان بازگشت} = 11 - 2 = 9$$

$$\text{میانگین زمان بازگشت} = \frac{3+10+7+11+9}{5} = \frac{40}{5} = 8$$

زمان اجرای فرآیند - زمان بازگشت فرآیند = زمان انتظار فرآیند

$$P_1 \text{ زمان انتظار} = 3 - 2 = 1$$

$$P_2 \text{ زمان انتظار} = 10 - 3 = 7$$

$$P_3 \text{ زمان انتظار} = 7 - 2 = 5$$

$$P_4 \text{ زمان انتظار} = 11 - 3 = 8$$

$$P_5 \text{ زمان انتظار} = 9 - 2 = 7$$

$$\text{میانگین زمان انتظار} = \frac{1+7+5+8+7}{5} = \frac{28}{5} = 5.6$$

$$\text{میانگین زمان اجرا} = \frac{2+3+2+3+2}{5} = \frac{12}{5} = 2.4$$

میانگین زمان انتظار + میانگین زمان اجرا = میانگین زمان بازگشت

$$8 = 2.4 + 5.6$$

با توجه به اطلاعات به دست آمده، جدول قبل، به شکل زیر تکمیل می‌گردد:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P ₁	0	2	1	3
P ₂	0	3	7	10
P ₃	1	2	5	7
P ₄	1	3	8	11
P ₅	2	2	7	9

$$\text{میانگین زمان بازگشت} = \text{میانگین زمان انتظار} + \text{میانگین زمان اجرا}$$

$$8 = 5.6 + 2.4$$

۲- گزینه (۱) صحیح است.

مفهومی که از فرآیند و job در این تست مطرح شده غلط است. فرآیند دارای job نیست بلکه می‌تواند دارای تعدادی نخ (thread) باشد. طراح محترم، طبق کدام مرجع شما job را بخشی از فرآیند می‌دانید؟ اگر درس سیستم عامل مدنظر شماست، در این درس هر یک مدت یکبار زمان‌بند

بلند مدت یک job را از هارد انتخاب می‌کند و در RAM قرار می‌دهد تا درجه چند برنامه‌نگی کنترل گردد و این job موجود در RAM، فرآیند نام می‌گیرد.
طراح محترم آیا شما زمان پاسخ را سلیقه‌ای تعریف می‌کنید؟ مگر می‌شود تعاریف اساسی و بنیادی را سلیقه‌ای تعریف نمود؟! در کدام مرجع زمان پاسخ این چنین که شما بیان کرده‌اید، تعریف شده است؟!

جهت اطلاع: تعریف درست زمان پاسخ: از لحظه‌ی ورود فرآیند تا نمایش اولین خروجی.
تعریف درست زمان گردش کار (بازگشت): از لحظه‌ی ورود فرآیند تا نمایش آخرین خروجی (اتمام کار). هیچ مرجعی نگفته است از لحظه‌ی ورود کار تا لحظه‌ی اولین دریافت cpu آیا بهتر نبود به جای نقض، دستکاری و خرابکاری تعاریف بنیادی بر اساس مفاهیم اصلی سوال طرح می‌شد؟ حداقل بهتر بود، لحظه‌ی ورود اولین کارها را مشخص می‌کردید.
به هر حال...

اگر فرض کنیم P_1 و P_2 در ابتدا با هم وارد می‌شوند. می‌توان job های دو فرآیند و زمان ورود آنها را با توجه به فرضیات سوال به صورت زیر در نظر گرفت:

زمان اجرا	زمان ورود	jobهای فرآیند P_1
3	0	J_{11}
3	5	J_{12}
3	10	J_{13}
3	15	J_{14}

زمان اجرا	زمان ورود	jobهای فرآیند P_2
1	0	J_{21}
1	4	J_{22}
1	8	J_{23}
1	12	J_{24}
1	16	J_{25}

در صورت سوال بیان شده است که اولویت با job قبلی موجود در سیستم است که تازه پردازنده را رها کرده است.

از آنجایی که کارهای مربوط به فرآیندهای P_1 و P_2 به ترتیب به صورت دوره‌ای در هر 5 ثانیه و 4 ثانیه یکبار شروع می‌شود، بنابراین به صورت دوره‌ای هر 20 ثانیه یکبار دو فرآیند P_1 و P_2 همزمان با هم وارد سیستم شده و روند اجرای آنها از سر گرفته می‌شود، پس جهت محاسبه‌ی بهره‌وری پردازنده و میانگین زمان پاسخ کافی است که اجرای کارهای فرآیند P_1 و P_2 را در یک دوره‌ی زمانی از صفر تا 20 ثانیه مورد بررسی قرار دهیم.

برای اینکه ببینیم اولین یا کوچکترین عددی که هم بر 4 و هم بر 5 بخش پذیر است، به عبارت دیگر برای محاسبه‌ی کوچکترین مضرب مشترک دو عدد 4 و 5 باید (ک.م.م) را محاسبه نمود:

$$A \cup B = \frac{A \times B}{A \cap B} \quad \text{ک.م.م} = \frac{4 \times 5}{\text{ب.م.م}}$$

بنابراین داریم:

	1	4	
5	4	1	0
4	4		

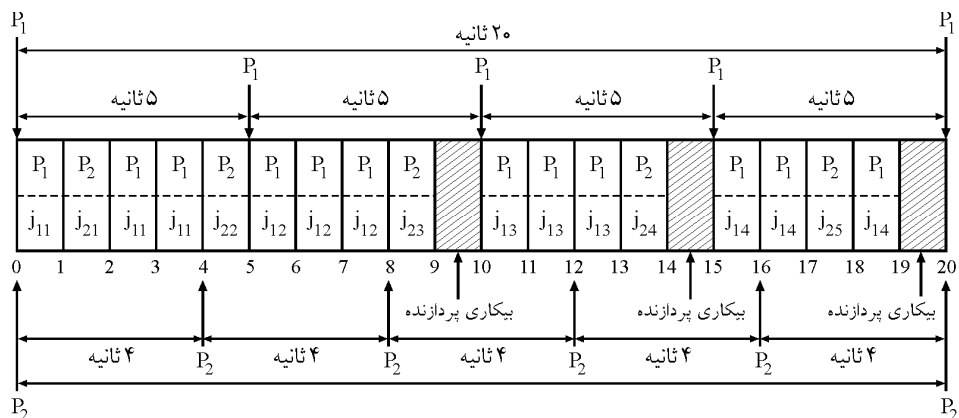
ب.م.م

توجه: ب.م.م، بزرگترین مقسوم علیه مشترک می‌باشد.

$$\text{ب.م.م} = \frac{4 \times 5}{1} = 20$$

توجه: ب.م.م دو عدد متوالی (پشت سر هم) همواره برابر یک است.

توجه: ک.م.م دو عدد متوالی (پشت سر هم) همواره برابر حاصل ضرب آن دو عدد است.



از آنجا که پردازنده از 20 ثانیه، 17 ثانیه را مشغول اجرا بوده است و 3 ثانیه را مشغول اجرا نبوده است، پس بهره‌وری آن معادل 85 درصد است.

$$\text{بهره‌وری پردازنده} = \frac{\text{زمان مفید پردازنده}}{\text{زمان مصرفی پردازنده}} = \frac{17}{20} = 0.85 \times 100\% = 85\%$$

و یا با توجه به دوره تناوب و زمان اجرای مطرح شده برای هر کار، بهره‌وری پردازنده به صورت مجموع بهره‌وری کارها و به شرح زیر قابل محاسبه است: (در سیستم‌های متناوب این فرمول کاربرد دارد).

$$U_{cpu} = \sum_{i=1}^m \frac{C_i}{P_i} = \frac{C_1}{P_1} + \frac{C_2}{P_2} = \frac{3}{5} + \frac{1}{4} = 0.6 + 0.25 = 0.85 \times 100\% = 85\%$$

C_i : زمان اجرای کار

P_i : دوره تناوب

در دوره‌ی زمانی صفر تا 20 ثانیه، کارهای فرآیند P₁، 4 بار و کارهای فرآیند P₂، 5 بار، یعنی در مجموع 9 کار وارد سیستم شده است.
تعریف زمان پاسخ از نگاه طراح: تأخیر بین ورود هر کار و اولین زمان در اختیار گرفتن پردازنده توسط آن کار.

زمان ورود کار - زمان دریافت پردازنده = زمان پاسخ کار

برای فرآیند P₁ داریم:

$$\begin{aligned} J_{11} \text{ زمان پاسخ} &= 0 - 0 = 0 \\ J_{12} \text{ زمان پاسخ} &= 5 - 5 = 0 \\ J_{13} \text{ زمان پاسخ} &= 10 - 10 = 0 \\ J_{14} \text{ زمان پاسخ} &= 15 - 15 = 0 \end{aligned}$$

برای فرآیند P₂ داریم:

$$\begin{aligned} J_{21} \text{ زمان پاسخ} &= 1 - 0 = 1 \\ J_{22} \text{ زمان پاسخ} &= 4 - 4 = 0 \\ J_{23} \text{ زمان پاسخ} &= 8 - 8 = 0 \\ J_{24} \text{ زمان پاسخ} &= 13 - 12 = 1 \\ J_{25} \text{ زمان پاسخ} &= 17 - 16 = 1 \end{aligned}$$

$$\text{میانگین زمان پاسخ} = \frac{(0+0+0+0)+(1+0+0+1+1)}{9} = \frac{3}{9} = \frac{1}{3}$$

۳- گزینه (۲) صحیح است.

جهت زمان‌بندی فرآیندها در الگوریتم SPN(SJF) می‌بایست قبل از اجرای فرآیندها، از زمان اجرای هر یک اطلاع داشته باشیم، در غیر این صورت با استفاده از فرمول تخمینی زیر (الگوریتم سالمندی (Aging))، از این الگوریتم می‌توانیم بهره ببریم:
 زمان اجرای واقعی پدیده‌ی لحظه‌ای + تخمین قدیم = تخمین جدید

$$S_{n+2} = \alpha S_{n+1} + (1-\alpha)T_{n+1}$$

S_{n+2} : زمان اجرای تخمین نوبت n+2

S_{n+1} : زمان اجرای تخمین نوبت n+1

T_{n+1} : زمان اجرای واقعی نوبت n+1

T_{n+2} : زمان اجرای واقعی نوبت n+2

α ($0 < \alpha < 1$): نشان می‌دهد که برای پدیده‌ی لحظه‌ای که الان اتفاق افتاده است، برای T_{n+1} چه وزنی را نسبت به داده‌های قبلی قائل هستیم.

توجه: هر چه α بیشتر باشد، اهمیت پدیده‌ی لحظه‌ای کمتر می‌شود.

مثال: از شما می‌پرسند معدل کل شما در 6 ترم گذشته چند می‌شود؟

معدل این ترم 16 و میانگین 5 ترم گذشته 15.5

معدل پدیده‌ی لحظه‌ای + تخمین قدیم = تخمین جدید

$$\text{معدل کل} = \frac{5}{6} \times 15.5 + \frac{1}{6} \times 16$$

چون اهمیت ترم ششم، $\frac{1}{6}$ است، بنابراین به این ترم نسبت به 5 ترم گذشته وزن می‌دهیم.

یعنی به پدیده‌ی قدیمی وزن $\frac{5}{6}$ و به پدیده‌ی جدید وزن $\frac{1}{6}$ را می‌دهیم.

بنابراین به کمک فرمول فوق و مقدار $\alpha = \frac{1}{2}$ ، زمان برآورده شده برای مرحله‌ی $n+2$ ام را برای هر یک از وظایف به دست می‌آوریم:

$$S_{n+2} = \alpha S_{n+1} + (1-\alpha)T_{n+1}$$

$$S_{n+2} = \frac{1}{2} S_{n+1} + \frac{1}{2} T_{n+1}$$

وظایف	زمان ورود	S_{n+1}	T_{n+1}	S_{n+2}	T_{n+2}
T_1	0	4	2	$\frac{1}{2} \times 4 + \frac{1}{2} \times 2 = 3$	4
T_2	0	6	4	$\frac{1}{2} \times 6 + \frac{1}{2} \times 4 = 5$	3
T_3	3	6	6	$\frac{1}{2} \times 6 + \frac{1}{2} \times 6 = 6$	6

از ستون S_{n+2} در جدول فوق به این نتیجه می‌رسیم که وظیفه T_1 که دارای زمان اجرای کوچکتر از زمان اجرای وظیفه‌ی T_2 می‌باشد، پس در زمان‌بندی اولویت می‌گیرد و اول وظیفه T_1 باید اجرا گردد، سپس T_2 و در آخر وظیفه‌ی T_3 اجرا شود.

SJF	<table border="1" style="display: inline-table; text-align: center;"> <tr> <td style="padding: 2px 10px;">T_1</td> <td style="padding: 2px 10px;">T_2</td> <td style="padding: 2px 10px;">T_3</td> </tr> <tr> <td style="padding: 2px 10px;">0</td> <td style="padding: 2px 10px;">4</td> <td style="padding: 2px 10px;">7</td> </tr> <tr> <td style="padding: 2px 10px;"></td> <td style="padding: 2px 10px;"></td> <td style="padding: 2px 10px;">13</td> </tr> </table>	T_1	T_2	T_3	0	4	7			13
T_1	T_2	T_3								
0	4	7								
		13								

توجه: در نمودار زمانی گانت فوق، مقادیر زمان واقعی T_{n+2} باید در نظر گرفته شود.

$$\text{میانگین زمان کامل (زمان گردش کار)} = \frac{(4-0) + (7-0) + (13-3)}{3} = \frac{21}{3} = 7$$

در پیاده‌سازی الگوریتم RR و FCFS، نیازی به دانستن و یا محاسبه‌ی زمان اجرای فرآیندها قبل از اجرا نمی‌باشد. زیرا FCFS بر اساس خروج به ترتیب ورود می‌باشد و RR، همانند FCFS می‌باشد که می‌چرخد. بنابراین داریم:

FCFS	T ₁	T ₂	T ₃
	0	4	7
			13

$$\text{میانگین زمان کامل (زمان گردش کار)} = \frac{(4-0) + (7-0) + (13-3)}{3} = \frac{21}{3} = 7$$

توجه: در صورت مسأله برش زمانی ذکر نشده است. بنابراین امکان محاسبه‌ی میانگین زمان کامل به کمک الگوریتم RR میسر نمی‌باشد.

۴- گزینه () صحیح است.

تخصیص منابع به هر فرآیند را مطابق نمودار ذیل انجام می‌دهیم:

	0	1	6	11	15	16	18											
P ₁	CPU	3	7	I/O	9	RDY	CPU	33	34	36								
P ₂											RDY	CPU	Wait	10	15	I/O	19	22
P ₃				RDY		CPU												
P ₄											40	41	46	47	51	53	54	55
											CPU	CPU	I/O	CPU				

توجه: منظور از RDY در نمودار فوق در صف آماده (Ready) می‌باشد.

$$\text{واحد زمانی} = \frac{1}{4} [(18-0) + (36-3) + (22-7) + (55-40)] = 20.25$$

حاصل جمع زمان‌های اجرای فرآیندها:

$$\underbrace{(\text{CPU}5 + \text{CPU}2)}_{P_1} + \underbrace{(\text{CPU}2 + \text{CPU}2)}_{P_2} + \underbrace{(\text{CPU}8)}_{P_3} + \underbrace{(\text{CPU}9 + \text{CPU}1)}_{P_4} = 29$$

$$\text{حاصل جمع زمان‌های ورودی و خروجی فرآیندها} = \underbrace{(\text{IO}5)}_{P_1} + \underbrace{(\text{IO}22)}_{P_2} + \underbrace{(\text{IO}2)}_{P_4} = 29$$

$$\text{بهره‌وری ورودی و خروجی} = \frac{\text{حاصل جمع زمان‌های ورودی و خروجی فرآیندها}}{\text{زمان کل}} \times 100$$

$$= \frac{29}{53} \times 100 = 54.71\%$$

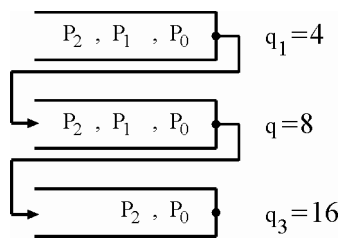
$$\text{بهره‌وری پردازنده} = \frac{\text{حاصل جمع زمان‌های اجرای فرآیندها}}{\text{زمان کل}} \times 100 = \frac{29}{55} \times 100 = 52.72\%$$

توجه: سازمان سنجش آموزش کشور در کلید اولیه خود ابتدا گزینه چهارم را به عنوان پاسخ اعلام نمود، سپس در کلید نهایی نظر خود را عوض کرد و کلاً تست را حذف نمود، که عمل درستی را انجام داده است که البته عمل درست‌تر آن است که سؤال از ابتدا، درست طرح گردد.

۵- گزینه () صحیح است.

با توجه به مفروضات مطرح شده در صورت سؤال داریم:

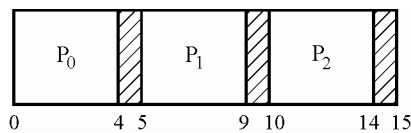
فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P_0	0	15		
P_1	0	12		
P_2	0	25		



ابتدا فرآیندها در صف اول با تکه زمانی (کوانتوم زمانی) معادل 4 ثانیه قرار می‌گیرند. و اگر در مدت 4 ثانیه تمام نشوند به صف دوم با تکه زمانی (کوانتوم زمانی) معادل 8 ثانیه منتقل می‌گردند. و اگر باز هم در مدت 8 ثانیه تمام نشوند به صف سوم با تکه زمانی (کوانتوم زمانی) معادل 16 ثانیه منتقل می‌گردند، تا بالاخره تمام شوند.

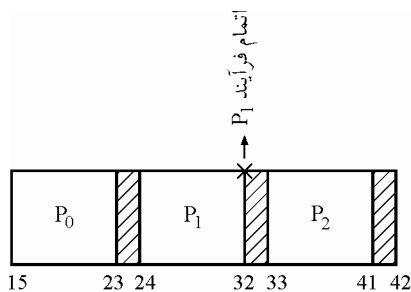
نمودار گانت این فرآیندها در هر صف به صورت زیر است:

صف اول: $TS=4s$



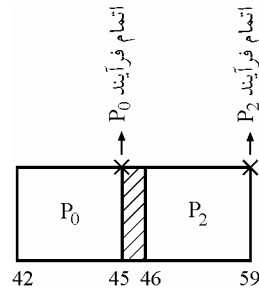
در صف اول، از فرآیندهای P_0, P_1, P_2 به ترتیب، 11، 8 و 21 ثانیه باقی می‌ماند. بنابراین این فرآیندها به صف دوم با تکه زمانی معادل 8 ثانیه منتقل می‌شوند.

صف دوم: $TS=8s$



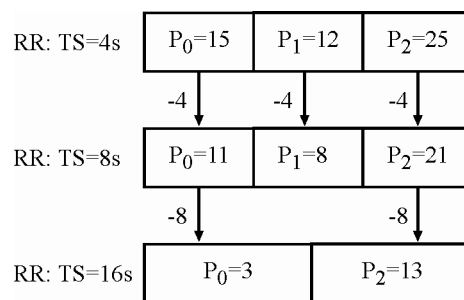
در صف دوم، فرآیند P_1 در لحظه 32 خاتمه می‌یابد و از فرآیندهای P_0 و P_2 به ترتیب، 3 و 13 ثانیه باقی می‌ماند. بنابراین این فرآیندها به صف سوم با تکه زمانی معادل 16 ثانیه منتقل می‌شوند، تا بالاخره تمام شوند.

صف سوم: $TS=16s$

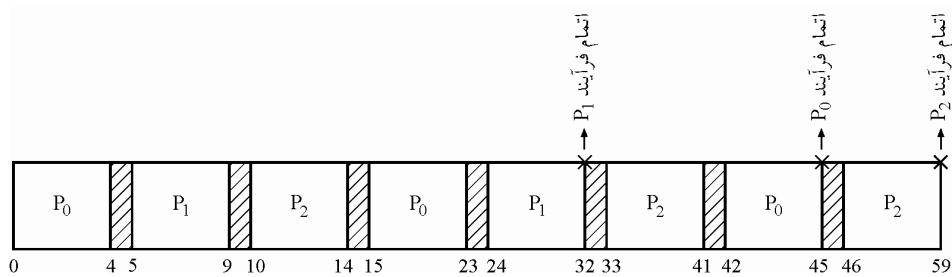


در صف سوم، فرآیندهای P_0 و P_2 در لحظات 45 و 59 خاتمه می‌یابند.

مدل‌سازی صف‌های فوق به صورت زیر است:



نمودار گانت کلی صف‌های فوق به صورت زیر است:



زمان ورود فرآیند - زمان خروج فرآیند = زمان بازگشت فرآیند

$$P_0 \text{ زمان بازگشت} = 45 - 0 = 45$$

$$P_1 \text{ زمان بازگشت} = 32 - 0 = 32$$

$$P_2 \text{ زمان بازگشت} = 59 - 0 = 59$$

$$\text{میانگین زمان بازگشت} = \frac{45 + 32 + 59}{3} = \frac{136}{3} = 45.33$$

زمان اجرای فرآیند - زمان بازگشت فرآیند = زمان انتظار فرآیند

$$P_0 \text{ زمان انتظار} = 45 - 15 = 30$$

$$P_1 \text{ زمان انتظار} = 32 - 12 = 20$$

$$P_2 \text{ زمان انتظار} = 59 - 25 = 34$$

$$\text{میانگین زمان انتظار} = \frac{30 + 20 + 34}{3} = \frac{84}{3} = 28$$

$$\text{میانگین زمان اجرا} = \frac{15 + 12 + 25}{3} = \frac{52}{3} = 17.33$$

میانگین زمان انتظار + میانگین زمان اجرا = میانگین زمان بازگشت

$$45.33 = 17.33 + 28$$

با توجه به اطلاعات به دست آمده، جدول قبل، به شکل زیر تکمیل می‌گردد:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P ₀	0	15	30	45
P ₁	0	12	20	32
P ₂	0	25	34	59

$$\text{میانگین زمان بازگشت} = \text{میانگین زمان انتظار} + \text{میانگین زمان اجرا}$$

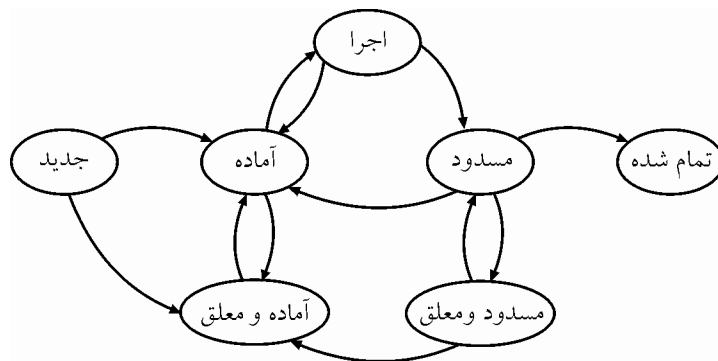
$$45.33 = 28 + 17.33$$

توجه: سازمان سنجش آموزش کشور در کلید اولیه خود ابتدا گزینه دوم را به عنوان پاسخ اعلام نمود، سپس در کلید نهایی نظر خود را عوض کرد و کلاً تست را حذف نمود، که عمل درستی را انجام داده است که البته عمل درست‌تر آن است که سؤال از ابتدا، درست طرح گردد.

۶- گزینه (۳) صحیح است.

بیشتر سیستم عامل‌ها حالت‌های فرآیند را فقط محدود به پنج حالت جدید، آماده، اجرا، مسدود و پایان یافته نمی‌کنند، بلکه یک موقعیت جدید با عنوان معلق (suspend) نیز برای فرآیند در نظر می‌گیرند. در حالت معلق فرآیند از حافظه به روی دیسک بیرون رانده شده است. به عنوان مثال فرض کنید همه‌ی فرآیندهای موجود در سیستم به حالت مسدود نقل مکان کنند (و همگی مثلاً

منتظر اتمام ورودی و خروجی مختص به خود هستند)، بنابراین هیچ فرآیندی در صف آماده قرار ندارد. در این حالت سیستم عامل جهت بیکار نماندن cpu، می تواند یک فرآیند را که در حالت مسدود است، از حافظه به دیسک منتقل کرده و یک فرآیند جدید را وارد صف آماده کند. به شکل زیر دقت کنید:



زمان‌بند میان مدت (Middle Term Scheduler) بنا به دلایلی ممکن است فرآیندهایی را از روی حافظه، به دیسک و بالعکس منتقل کند. در واقع وظیفه‌ی این زمان‌بند جابه‌جا کردن فرآیندها بین حافظه و دیسک می‌باشد. به این کار مبادله (Swapping) گفته می‌شود. بنابراین زمان‌بند میان مدت (سطح میانی) که همان زمان‌بند حافظه یا swapper است، وظیفه suspend کردن (تعطیل) و دوباره فعال‌سازی (آماده کردن) فرآیندها را برعهده دارد.

۷- گزینه (۴) صحیح است.

فرآیند	زمان ورود	زمان پردازش
A	0	$t_A = 3$
B	1	t_B
C	2	t_C

در الگوریتم FCFS

در این روش، فرآیندها به همان ترتیب ورود به سیستم، در یک صف قرار گرفته و از ابتدای صف به ترتیب، پردازنده را در اختیار می‌گیرند.

توجه: FCFS یک الگوریتم انحصاری است.

با توجه به مفروضات مسأله، نمودار گانت زیر را داریم:

A	B	C
0	3	$3+t_B$
		$3+t_B+t_C$

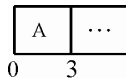
در الگوریتم HRRN

برای تعیین اولویت یک فرآیند در الگوریتم HRRN از فرمول زیر استفاده می‌شود:

$$اولویت = 1 + \frac{\text{زمان انتظار}}{\text{زمان اجرا}} = \frac{\text{زمان انتظار} + \text{زمان اجرا}}{\text{زمان اجرا}}$$

نکته: HRRN یک الگوریتم انحصاری است.

در این الگوریتم در زمان صفر فقط فرآیند A قرار دارد که به شکل انحصاری اجرا می‌گردد.



اما در زمان 3 فرآیندهای B و C در صف آماده قرار دارند. انتظار هر یک از فرآیندهای B و C تا لحظه 3 به صورت زیر است:

$$B \text{ انتظار} = 3 - 1 = 2$$

$$C \text{ انتظار} = 3 - 2 = 1$$

در ادامه براساس رابطه اولویت داریم:

$$اولویت (B) = \frac{2}{t_B} + 1$$

$$اولویت (C) = \frac{1}{t_C} + 1$$

حال اگر مطابق فرض مسأله قرار باشد، زمان‌بندی‌های دو الگوریتم FCFS و HRRN یکسان باشد. پس در الگوریتم HRRN هم باید اول فرآیند A اجرا گردد که بدیهی است و اجرا می‌گردد، اما بعد از اجرای فرآیند A، در لحظه 3، دو فرآیند B و C در الگوریتم HRRN شانس اجرا دارند، ولی از آنجا که باید این الگوریتم مانند FCFS باشد، پس باید اول فرآیند B اجرا گردد و بعد، فرآیند C و این زمانی رخ می‌دهد که اولویت فرآیند B از اولویت فرآیند C بیشتر باشد، یعنی داشته باشیم:

$$اولویت (B) > اولویت (C)$$

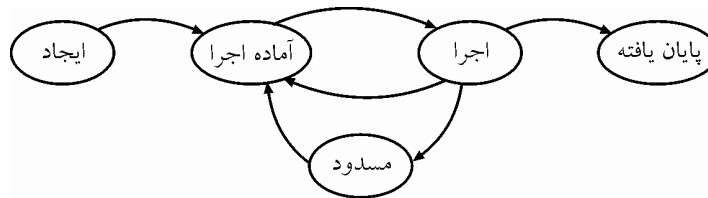
$$\frac{2}{t_B} + 1 > \frac{1}{t_C} + 1 \rightarrow$$

$$\frac{2}{t_B} > \frac{1}{t_C} \rightarrow t_B < 2t_C$$

تست جالبی بود.

۸- گزینه (۲) صحیح است.

تغییر حالات ممکن برای یک فرآیند مطابق حالت‌های ایجاد، مسدود، اجرا، پایان یافته و آماده اجرا مطابق شکل زیر می‌باشد:



بنابراین واضح است که تغییر حالتی از آماده اجرا به مسدود وجود ندارد، زیرا ابتدا باید به حالت اجرا برویم تا بتوانیم درخواستی بدهیم و بعد مسدود شویم.

۹- گزینه (۴) صحیح است.

الگوریتم FCFS (First Come First Served)

این الگوریتم ساده‌ترین الگوریتم زمان‌بندی پردازنده است. در این روش فرآیندها با همان ترتیب ورود به سیستم، در یک صف قرار گرفته و از ابتدای صف به ترتیب، پردازنده را در اختیار می‌گیرند.

توجه: این الگوریتم، FIFO (First In First Out) نیز نامیده می‌شود.

توجه: FCFS، یک الگوریتم انحصاری (Non Preemptive) است.

توجه: الگوریتم FIFO، مشکل قحطی‌زدگی ندارد.

الگوریتم RR (Round Robin)

الگوریتم RR (نوبت چرخشی) یکی از پرکاربردترین الگوریتم‌ها در سیستم‌های اشتراک زمانی است. درباره نحوه عملکرد آن می‌توان گفت این الگوریتم نسخه غیرانحصاری (Preemptive) الگوریتم FCFS می‌باشد. در این الگوریتم، زمان پردازنده را به برش‌های زمانی کوتاه (Time Slice) تقسیم می‌کنیم. همانند الگوریتم FCFS، فرآیندهایی که به سیستم تحویل داده می‌شوند به انتهای یک صف وارد می‌شوند. سپس پردازنده از ابتدای صف شروع کرده و به هر فرآیند حداکثر به اندازه یک برش زمانی سرویس می‌دهد.

در واقع پس از آنکه برش زمانی یک فرآیند به پایان رسید، پردازنده آن فرآیند را رها کرده و به سراغ فرآیند بعدی موجود در صف می‌رود. این عمل آنقدر تکرار می‌شود تا پردازنده به انتهای صف فرآیندهای آماده برسد. به عبارت دیگر فرآیندها در یک صف دایره‌ای شکل سازماندهی می‌شوند و پردازنده به صورت دوار در این صف حرکت کرده و به هر فرآیند فقط به اندازه حداکثر یک برش زمانی سرویس می‌دهد.

توجه: به برش زمانی، کوانتوم زمانی (Quantum Time) نیز می‌گویند.

توجه: الگوریتم RR مشکل قحطی‌زدگی ندارد.

حال اگر ترتیب ورود پردازنده‌ها (فرآیند) یکسان در نظر گرفته شود و زمان اجرای پردازنده‌ها کوچکتر از زمان کوانتوم باشد، آنگاه این شرایط فراهم می‌گردد تا زمان‌بند RR دقیقاً مانند زمان‌بند FCFS

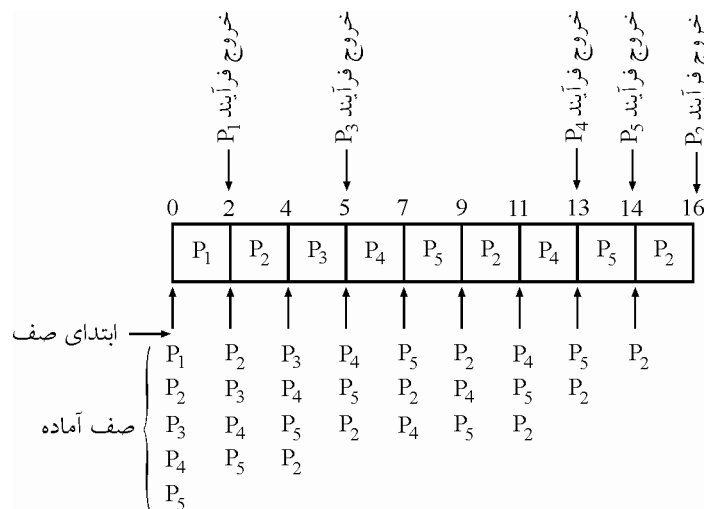
عمل نماید. چون در چنین شرایطی در زمان‌بند RR هر فرآیند قبل از پایان کوانتوم خاتمه می‌یابد و این یعنی FIFO.

۱۰- گزینه (۳) صحیح است.

با توجه به مفروضات مطرح شده در صورت سؤال داریم:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار+	زمان بازگشت =
P ₁	0	2		
P ₂	0	6		
P ₃	0	1		
P ₄	0	4		
P ₅	0	3		

با توجه به برش زمانی quantum = 2، نمودار گانت زیر را داریم:



زمان ورود فرآیند - زمان خروج فرآیند = زمان بازگشت فرآیند

$$P_1 \text{ زمان بازگشت} = 2 - 0 = 2$$

$$P_2 \text{ زمان بازگشت} = 16 - 0 = 16$$

$$P_3 \text{ زمان بازگشت} = 5 - 0 = 5$$

$$P_4 \text{ زمان بازگشت} = 13 - 0 = 13$$

$$P_5 \text{ زمان بازگشت} = 14 - 0 = 14$$

زمان اجرای فرآیند - زمان بازگشت فرآیند = زمان انتظار فرآیند

$$P_1 \text{ زمان انتظار} = 2 - 2 = 0$$

$$P_2 \text{ زمان انتظار} = 16 - 6 = 10$$

$$P_3 \text{ زمان انتظار} = 5 - 1 = 4$$

$$P_4 \text{ زمان انتظار} = 13 - 4 = 9$$

$$P_5 \text{ زمان انتظار} = 14 - 3 = 11$$

بنابراین مطابق خواسته سوال، زمان انتظار پردازنده 2 و 4 به ترتیب برابر $P_2 = 10$ و $P_4 = 9$ می‌باشد. بنابراین گزینه سوم پاسخ سؤال است.

۱۱- گزینه (۳) صحیح است.

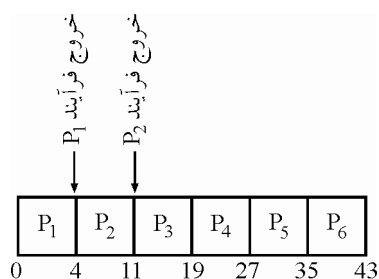
با توجه به مفروضات مطرح شده در صورت سؤال داریم:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P_1	0	4		
P_2	0	7		
P_3	0	12		
P_4	0	20		
P_5	0	25		
P_6	0	30		

ابتدا فرآیندها در صف اول با تکه زمانی معادل 8 میکروثانیه قرار می‌گیرند. و اگر در مدت 8 میکروثانیه تمام نشوند به صف دوم با تکه زمانی معادل 16 میکروثانیه منتقل می‌گردند. و اگر باز هم در مدت 16 میکروثانیه تمام نشوند به صف سوم با الگوریتم FCFS منتقل می‌گردند، تا بالاخره تمام شوند.

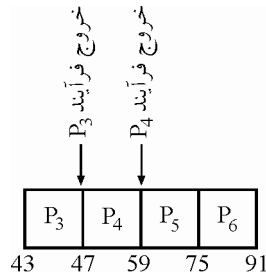
نمودار گانت این فرآیندها در هر صف به صورت زیر است:

صف اول: $TS = 8\mu s$



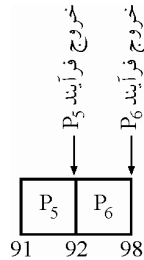
در صف اول، فرآیندهای P_1 و P_2 در لحظات 4 و 11 خاتمه می‌یابند و از فرآیندهای P_3 تا P_6 به ترتیب، 4، 12، 17 و 22 میکروثانیه باقی می‌ماند. بنابراین این فرآیندها به صف دوم با تکه زمانی معادل 16 میکروثانیه منتقل می‌شوند.

صف دوم: $TS = 16\mu s$



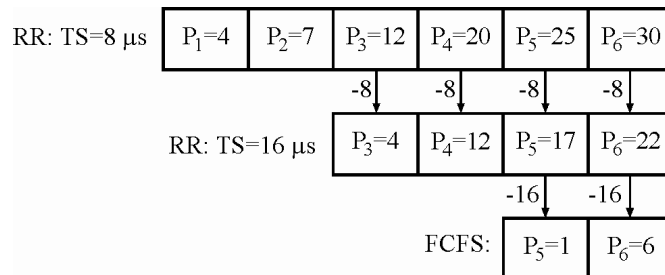
در صف دوم، فرآیندهای P_3 و P_4 در لحظات 47 و 59 خاتمه می‌یابند و از فرآیندهای P_5 و P_6 به ترتیب، 1 و 6 میکروثانیه باقی می‌ماند. بنابراین این فرآیندها به صف سوم با الگوریتم FCFS منتقل می‌شوند.

صف سوم: FCFS

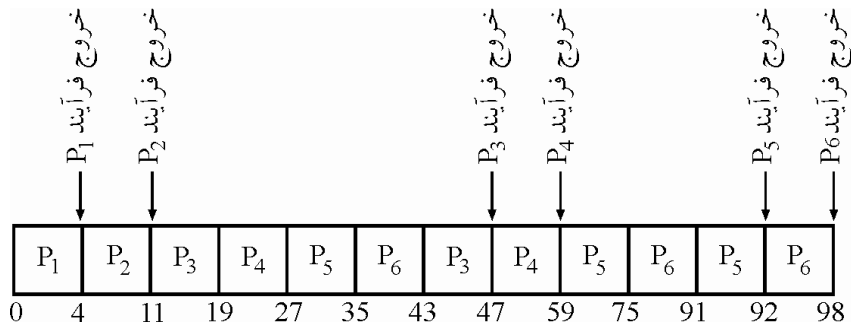


در صف سوم، فرآیندهای P_5 و P_6 در لحظات 92 و 98 خاتمه می‌یابند.

مدل‌سازی صف‌های فوق به صورت زیر است:



نمودار گانت کلی صف‌های فوق به صورت زیر است:



زمان ورود فرآیند - زمان خروج فرآیند = زمان بازگشت فرآیند

$$P_1 \text{ زمان بازگشت} = 4 - 0 = 4$$

$$P_2 \text{ زمان بازگشت} = 11 - 0 = 11$$

$$P_3 \text{ زمان بازگشت} = 47 - 0 = 47$$

$$P_4 \text{ زمان بازگشت} = 59 - 0 = 59$$

$$P_5 \text{ زمان بازگشت} = 92 - 0 = 92$$

$$P_6 \text{ زمان بازگشت} = 98 - 0 = 98$$

$$\text{میانگین زمان بازگشت} = \text{ATT} = \frac{4 + 11 + 47 + 59 + 92 + 98}{6} = \frac{311}{6} = 51.83$$

زمان اجرای فرآیند - زمان بازگشت فرآیند = زمان انتظار فرآیند

$$P_1 \text{ زمان انتظار} = 4 - 4 = 0$$

$$P_2 \text{ زمان انتظار} = 11 - 7 = 4$$

$$P_3 \text{ زمان انتظار} = 47 - 12 = 35$$

$$P_4 \text{ زمان انتظار} = 59 - 20 = 39$$

$$P_5 \text{ زمان انتظار} = 92 - 25 = 67$$

$$P_6 \text{ زمان انتظار} = 98 - 30 = 68$$

$$\text{میانگین زمان انتظار} = \text{AWT} = \frac{0 + 4 + 35 + 39 + 67 + 68}{6} = \frac{213}{6} = 35.5$$

$$\text{میانگین زمان اجرا} = \text{AST} = \frac{4 + 7 + 12 + 20 + 25 + 30}{6} = \frac{98}{6} = 16.33$$

$$\text{AVG Turnaround Time} = \text{AVG Service Time} + \text{AVG Waiting Time}$$

میانگین زمان انتظار + میانگین زمان اجرا = میانگین زمان بازگشت

$$51.83 = 16.33 + 35.5$$

توجه: مطابق رابطه فوق، تفاضل میانگین زمان بازگشت و میانگین زمان انتظار باید برابر میانگین زمان اجرا باشد، که فقط در گزینه سوم و چهارم این مورد رعایت شده است. بنابراین گزینه اول و دوم نادرست است.

توجه: همچنین مطابق رابطه فوق، میانگین زمان بازگشت همواره از میانگین زمان انتظار بیشتر است، بنابراین از این نگاه نیز گزینه اول نادرست است.

با توجه به اطلاعات به دست آمده، جدول قبل، به شکل زیر تکمیل می‌گردد:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P ₁	0	4	0	4
P ₂	0	7	4	11
P ₃	0	12	35	47
P ₄	0	20	39	59
P ₅	0	25	67	92
P ₆	0	30	68	98

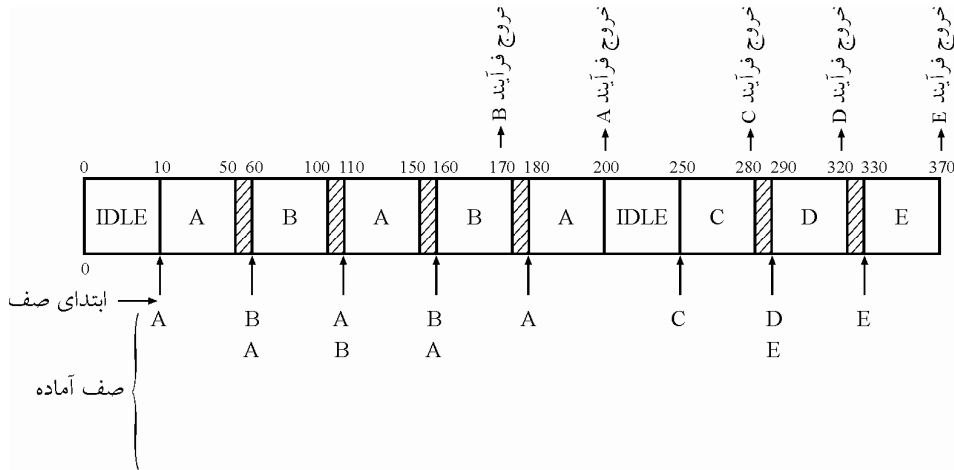
$$\begin{array}{rcl} \text{میانگین زمان بازگشت} & = & \text{میانگین زمان انتظار} + \text{میانگین زمان اجرا} \\ 51.83 & & 35.5 \quad 16.33 \end{array}$$

۱۲- گزینه (۲) صحیح است.

با توجه به مفروضات مطرح شده در صورت سؤال داریم:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
A	10	100		
B	20	50		
C	250	30		
D	260	30		
E	270	40		

با توجه به برش زمانی $40 = \text{quantum}$ و زمان سربرار تعویض متن فرآیندها $10 = \text{context switch}$ نمودار گانت زیر را داریم:



زمان ورود فرآیند - زمان خروج فرآیند = زمان بازگشت فرآیند

$$A \text{ زمان بازگشت} = 200 - 10 = 190$$

$$B \text{ زمان بازگشت} = 170 - 20 = 150$$

$$C \text{ زمان بازگشت} = 280 - 250 = 30$$

$$D \text{ زمان بازگشت} = 320 - 260 = 60$$

$$E \text{ زمان بازگشت} = 370 - 270 = 100$$

$$\text{میانگین زمان بازگشت} = \frac{190 + 150 + 30 + 60 + 100}{5} = \frac{530}{5} = 106$$

زمان اجرای فرآیند - زمان بازگشت فرآیند = زمان انتظار فرآیند

$$A \text{ زمان انتظار} = 190 - 100 = 90$$

$$B \text{ زمان انتظار} = 150 - 50 = 100$$

$$C \text{ زمان انتظار} = 30 - 30 = 0$$

$$D \text{ زمان انتظار} = 60 - 30 = 30$$

$$E \text{ زمان انتظار} = 100 - 40 = 60$$

$$\text{میانگین زمان انتظار} = \frac{90 + 100 + 0 + 30 + 60}{5} = \frac{280}{5} = 56$$

$$\text{میانگین زمان اجرا} = \frac{100 + 50 + 30 + 30 + 40}{5} = \frac{250}{5} = 50$$

میانگین زمان انتظار + میانگین زمان اجرا = میانگین زمان بازگشت

$$106 = 50 + 56$$

با توجه به اطلاعات به دست آمده، جدول قبل، به شکل زیر تکمیل می‌گردد:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
A	10	100	90	190
B	20	50	100	150
C	250	30	0	30
D	260	30	30	60
E	270	40	60	100

$$\text{میانگین زمان بازگشت} = \text{میانگین زمان انتظار} + \text{میانگین زمان اجرا}$$

$$106 = 56 + 50$$

۱۳- گزینه (۳) صحیح است.

الگوریتم SJF (Shortest Job First)

در این روش ابتدا کاری برای اجرا انتخاب می‌شود که از همه کوتاهتر باشد (زمان اجرای کمتری داشته باشد).

توجه: این الگوریتم، SPN (Shortest Process Next) و

SPT (Shortest Process Time) نیز نامیده می‌شود.

توجه: SJF یک الگوریتم انحصاری (Non Preemptive) است.

توجه: یک نقص عمده الگوریتم SJF این است که ممکن است باعث قحطی‌زدگی فرآیندهای طولانی شود. به این ترتیب که اگر همواره تعدادی فرآیند کوچک وارد سیستم شوند، اجرای فرآیندهای بزرگ به طور متناوب به تعویق می‌افتد. این روال حتی می‌تواند تا بینهایت ادامه یابد و هیچگاه نوبت به فرآیندهای بزرگ نرسد!!!!

توجه: در این روش اگر دو فرآیند مدت زمان اجرای برابر داشته باشند، بر اساس FCFS زمان‌بندی می‌شوند.

توجه: هدف الگوریتم SJF به حداقل رساندن میانگین زمان انتظار، میانگین زمان پاسخ و میانگین زمان گردش کار (بازگشت) فرآیندهاست.

توجه: در عمل نمی‌توان الگوریتم SJF را پیاده‌سازی کرد، زیرا سیستم عامل زمان اجرای فرآیندها را از قبل نمی‌داند و تنها کاری که می‌تواند انجام دهد این است که زمان اجرای فرآیندها را فقط حدس زده و به طور تقریبی بدست آورد.

الگوریتم SRT (Shortest Remaining Time)

این الگوریتم نسخه غیرانحصاری (Preemptive) الگوریتم SJF است. در این الگوریتم اگر حین اجرای یک فرآیند، فرآیندی وارد شود که زمان اجرای کوتاه‌تری داشته باشد، پردازنده را در اختیار می‌گیرد.

توجه: این الگوریتم، **SRPT** (Shortest Remaining Processing Time)،

و **SRTF** (Shortest Remaining Time First)

SRTN (Shortest Remaining Time Next) نیز نامیده می‌شود.

توجه: اگر لحظه ورود همه فرآیندها یکی باشد، الگوریتم **SRT** مشابه **SJF** عمل می‌کند.

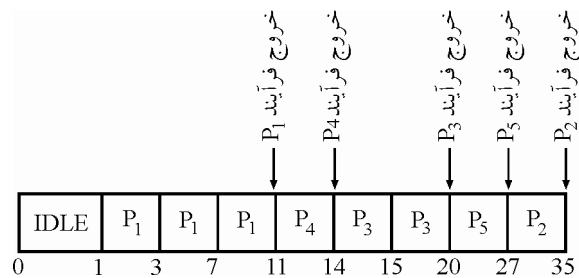
توجه: در الگوریتم **SRT** نیز همانند الگوریتم **SJF**، احتمال وقوع قحطی‌زدگی برای کارهای

بزرگ وجود دارد.

با توجه به مفروضات مطرح شده در صورت سؤال داریم:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P ₁	1	10		
P ₂	3	8		
P ₃	7	6		
P ₄	11	3		
P ₅	15	7		

با توجه به مفروضات مساله، نمودار گانت زیر را داریم:



زمان ورود فرآیند - زمان خروج فرآیند = زمان بازگشت فرآیند

$$P_1 \text{ زمان بازگشت} = 11 - 1 = 10$$

$$P_2 \text{ زمان بازگشت} = 35 - 3 = 32$$

$$P_3 \text{ زمان بازگشت} = 20 - 7 = 13$$

$$P_4 \text{ زمان بازگشت} = 14 - 11 = 3$$

$$P_5 \text{ زمان بازگشت} = 27 - 15 = 12$$

$$\text{میانگین زمان بازگشت} = \text{ATT} = \frac{10 + 32 + 13 + 3 + 12}{5} = \frac{70}{5} = 14$$

زمان اجرای فرآیند - زمان بازگشت فرآیند = زمان انتظار فرآیند

$$P_1 \text{ زمان انتظار} = 10 - 10 = 0$$

$$P_2 \text{ زمان انتظار} = 32 - 8 = 24$$

$$P_3 \text{ زمان انتظار} = 13 - 6 = 7$$

$$P_4 \text{ زمان انتظار} = 3 - 3 = 0$$

$$P_5 \text{ زمان انتظار} = 12 - 7 = 5$$

$$\text{میانگین زمان انتظار} = AWT = \frac{0 + 24 + 7 + 0 + 5}{5} = \frac{36}{5} = 7.2$$

$$\text{میانگین زمان اجرا} = AST = \frac{10 + 8 + 6 + 3 + 7}{5} = \frac{34}{5} = 6.8$$

$$AVG \text{ Turnaround Time} = AVG \text{ Service Time} + AVG \text{ Waiting Time}$$

$$\text{میانگین زمان بازگشت} = \text{میانگین زمان اجرا} + \text{میانگین زمان انتظار}$$

$$14 = 6.8 + 7.2$$

توجه: مطابق رابطه فوق، تفاضل میانگین زمان بازگشت و میانگین زمان انتظار باید برابر میانگین زمان اجرا باشد، که فقط در گزینه سوم این مورد رعایت شده است. بنابراین گزینه‌های اول، دوم و چهارم پاسخ سؤال نیستند.

توجه: همچنین مطابق رابطه فوق، میانگین زمان بازگشت همواره از میانگین زمان انتظار بیشتر است، بنابراین از این نگاه همه گزینه‌ها درست هستند، که در اینجا این نگاه کارآمد نیست.

با توجه به اطلاعات به دست آمده، جدول قبل، به شکل زیر تکمیل می‌گردد:

فرآیند	زمان ورود	زمان اجرا	زمان انتظار +	زمان بازگشت =
P ₁	1	10	0	10
P ₂	3	8	24	32
P ₃	7	6	7	13
P ₄	11	3	0	3
P ₅	15	7	5	12

$$\text{میانگین زمان بازگشت} = \text{میانگین زمان انتظار} + \text{میانگین زمان اجرا}$$

$$14 = 7.2 + 6.8$$