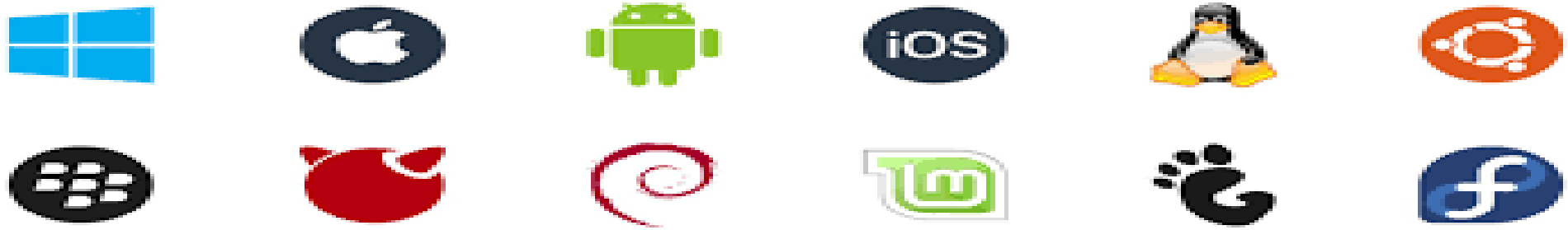


سیستم های عامل

Operating Systems

میلاد سلطانی

فصل چهارم

مدیریت حافظه

- سلسله مراتب حافظه
- تک برنامگی ساده و با Overlay
- سیستم مبادله ساده Swapping
- چند برنامگی با بخشی بندی ثابت حافظه
- آدرس فیزیکی، منطقی و تبدیل آدرس
- مبادله (Swapping) در سیستم چند برنامگی
- تخصیص چند بخشی همجوار
- مدیریت فضاهای آزاد
- روشهای تخصیص حافظه
- مشکلات مدیریت حافظه
  - ✓ تکه تکه شدن داخلی و خارجی
  - ✓ پراکندگی برنامه ها

مقدمه

- حافظه ایده آل از دید برنامه نویسان، حافظه ای بی نهایت بزرگ، سریع، ارزان و غیر فرّار است.

- بدلیل عدم وجود حافظه ایده آل، سیستم های کامپیوتری از یک سلسله مراتب حافظه استفاده می کنند.

- بخشی از سیستم عامل که سلسله مراتب حافظه را اداره می کند، به عنوان مدیر حافظه (Memory Manager) شناخته می شود.

سلسله مراتب حافظه

- سیستم های ذخیره سازی بر حسب قیمت و سرعت در یک سلسله مراتب سازماندهی می شوند.

سلسله مراتب حافظه

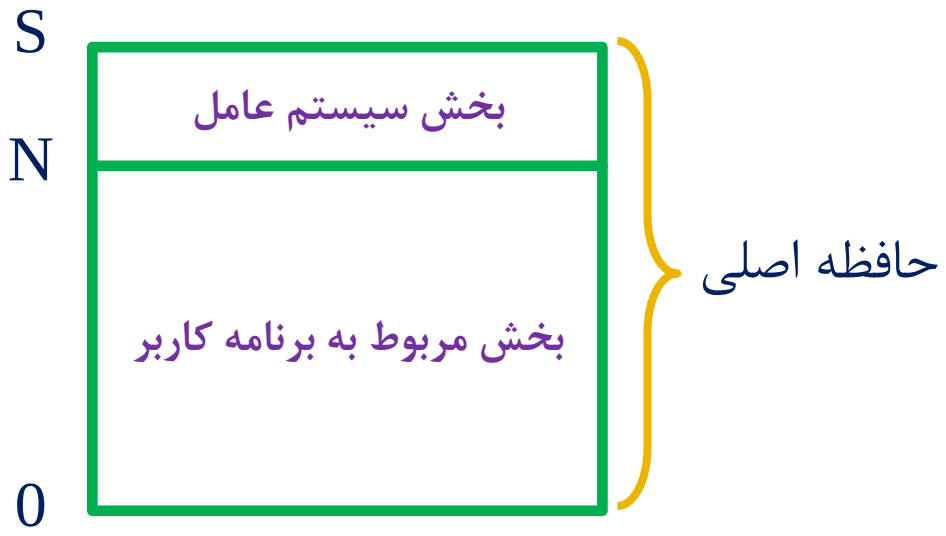
- حافظه اصلی
  - ✓ حافظه اصلی و ثبات های داخل CPU تنها ابزار ذخیره سازی اند که مستقیم توسط CPU دستکاری می شوند.
- دیسک های مغناطیسی (Magnetic Disks)
  - ✓ اطلاعات را با ثبت مغناطیسی روی دیسک نگهداری می کند.
- نوارهای مغناطیسی (Magnetic Tapes)
  - ✓ به دلیل کندی دسترسی معمولاً به عنوان رسانه پشتیبانی و انتقال داده ها استفاده می شود.
- حافظه نهان (Cache)
  - ✓ برای افزایش سرعت دسترسی به اطلاعات مورد نیاز استفاده می شود.

تک برنامگی ساده

- ساده ترین طرح مدیریت حافظه اینست که در هر لحظه فقط یک برنامه در حال اجرا برای استفاده از حافظه وجود داشته باشد.
- هنگام ورود برنامه به حافظه، کل حافظه در اختیار آن قرار می گیرد.
- در این مدیریت اگر حافظه به اندازه کافی موجود نباشد، برنامه اجرا نمی شود.
- سیستم عامل DOS نمونه این مدیریت می باشد.

تک برنامگی ساده

- در این سیستم بخشی از حافظه را خود سیستم عامل اشغال می کند.
- در این سیستم برنامه کاربر نباید آدرس بیشتر از N تولید کند ولی سیستم عامل به تمام حافظه دسترسی دارد.



تک برنامگی با Overlay

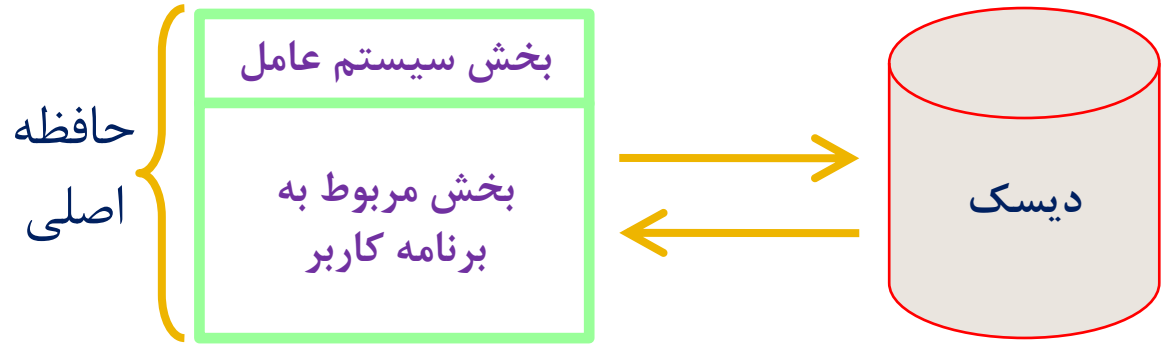
- این سیستم، بر خلاف سیستم تک برنامگی ساده، برنامه های بزرگتر از حافظه را می تواند اجرا کند.

- برنامه به بخش های مختلفی تقسیم می شود و تنها بخشی در حافظه قرار می گیرد که برای اجرا مورد نیاز است.

- این تکنیک در سیستم های عامل قدیمی مثل DOS کاربرد داشت ولی در سیستم های عامل جدید مثل ویندوز کاربرد ندارد.

سیستم مبادله ساده (Swapping)

- در این سیستم چند فرآیند قابل اجرا هستند ولی در هر لحظه فقط یک فرآیند در حافظه قرار می گیرد.
- برش زمانی هر فرآیند که به اتمام برسد، موقتاً به دیسک برده می شود و فرآیند جدید جایگزین می شود.



- حافظه مورد نیاز این سیستم برابر با اندازه سیستم عامل + اندازه فرآیند می باشد.

سیستم مبادله ساده (Swapping)

- زمان اجرای هر فرآیند بصورت زیر محاسبه می شود:

$$\text{زمان اجرا فرآیند} = \text{زمان مبادله} \times ۲ + \text{زمان CPU}$$

- بهره وری CPU نیز از رابطه زیر قابل محاسبه است:

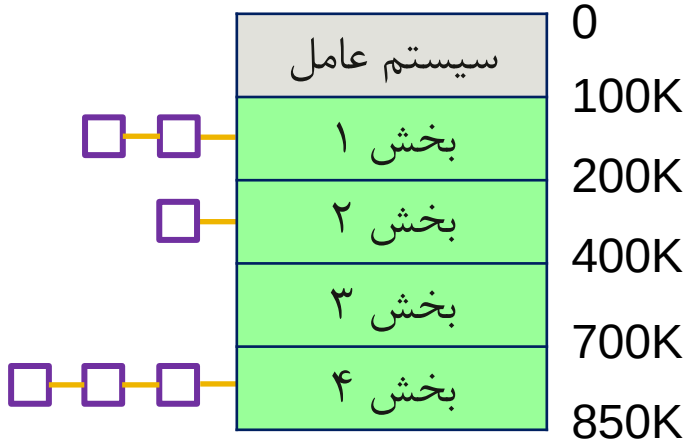
$$\text{درصد بکارگیری CPU} = \frac{\text{زمان CPU}}{\text{زمان CPU} + \text{زمان مبادله} \times ۲} \times ۱۰۰$$

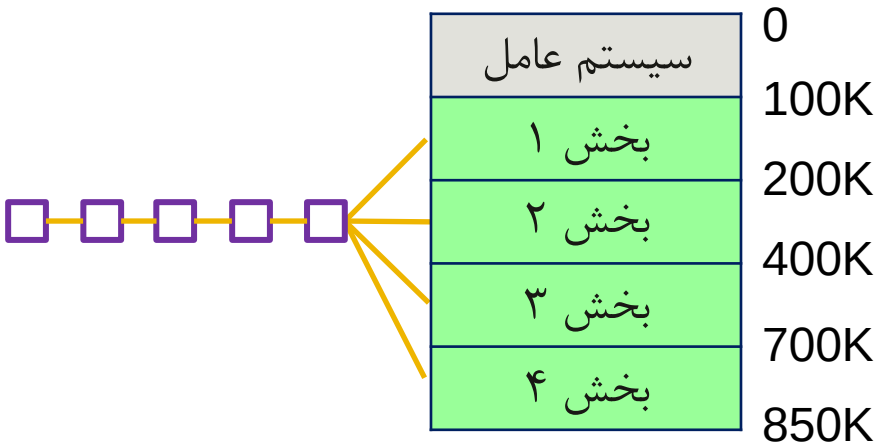
چند برنامگی با بخش بندی ثابت حافظه

- به این تکنیک، بخش بندی ایستا (Static) هم گفته می شود.
- ساده ترین راه چند برنامگی تقسیم کردن حافظه به N قسمت می باشد که بخش های ایجاد شده الزاماً هم اندازه نمی باشند.
- وقتی یک فرآیند وارد سیستم می شود در صف قرار گرفته تا در مناسب ترین بخشی که موجود است، قرار داده شود.
- این روش در سیستم های عامل IBM OS/360 روی کامپیوترهای بزرگ IBM اجرا می شد.
- این روش به دلیل اتلاف حافظه زیاد، در سیستم های عامل امروزی استفاده نمی شود.

چند برنامگی با بخش بندی ثابت حافظه

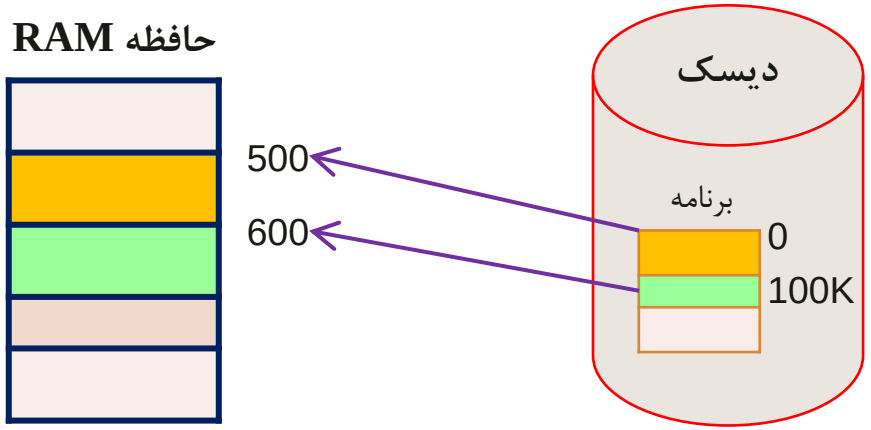
- این روش به دو صورت اجرا می شود :
  - ✓ استفاده از یک صف برای هر بخش
  - ✓ استفاده از یک صف برای همه بخش ها





آدرس منطقی و آدرس فیزیکی

- یک برنامه و اجزایش دارای دو گونه آدرس می باشند:
  - ✓ منطقی (Logical Address)
    - آدرس منطقی توسط CPU تولید شده و قابل درک توسط کاربر می باشد.
  - ✓ فیزیکی (Physical Address)
    - آدرس فیزیکی، مکان واقعی برنامه و اجزایش را در بستر حافظه تعیین می کند.

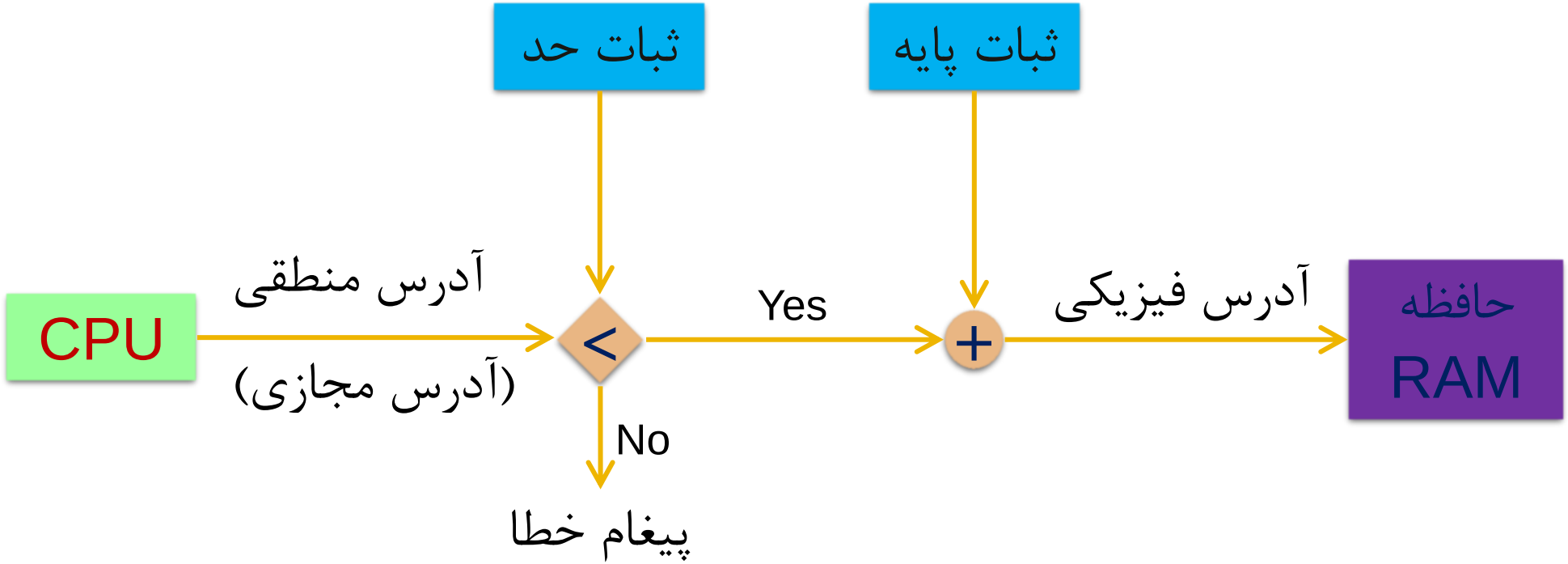


تبدیل آدرس

- در اصلاح به تبدیل آدرس، پیوند (Bind) گفته می شود که در واقع نگاشتی از یک فضای آدرس به فضای آدرس دیگر می باشد و به سه صورت انجام می گیرد:
  - ✓ پیوند آدرس زمان کامپایل
  - ✓ پیوند آدرس زمان بارگزاری
  - ✓ پیوند آدرس زمان اجرا
- اگر پیوند آدرس در زمان اجرا صورت گیرد، در این حالت به آدرس منطقی، آدرس مجازی (Virtual Address) گفته می شود.
- پیوند زمان اجرای آدرس های مجازی به فیزیکی توسط واحد سخت افزاری مدیریت حافظه (Memory Management Unit) MMU انجام می شود.

تبدیل آدرس

- یک طرح ساده از MMU بصورت زیر است:



مبادله در سیستم چند برنامگی (Swapping)

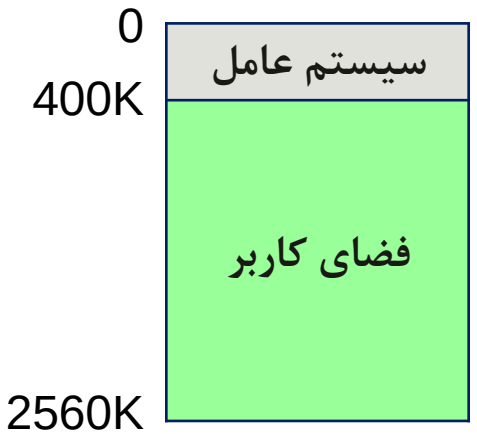
- این تکنیک مشابه با روش مبادله ساده (Swapping) می باشد.
- بهره وری CPU در این تکنیک از رابطه زیر قابل محاسبه است:

$$\text{درصد بکارگیری CPU} = \frac{\text{زمان CPU}}{\text{زمان CPU} + \text{زمان مبادله} \times ۲} \times ۱۰۰$$

تخصیص چند بخشی همجوار

- به این روش بخش بندی پویا (Dynamic) نیز گفته می شود.

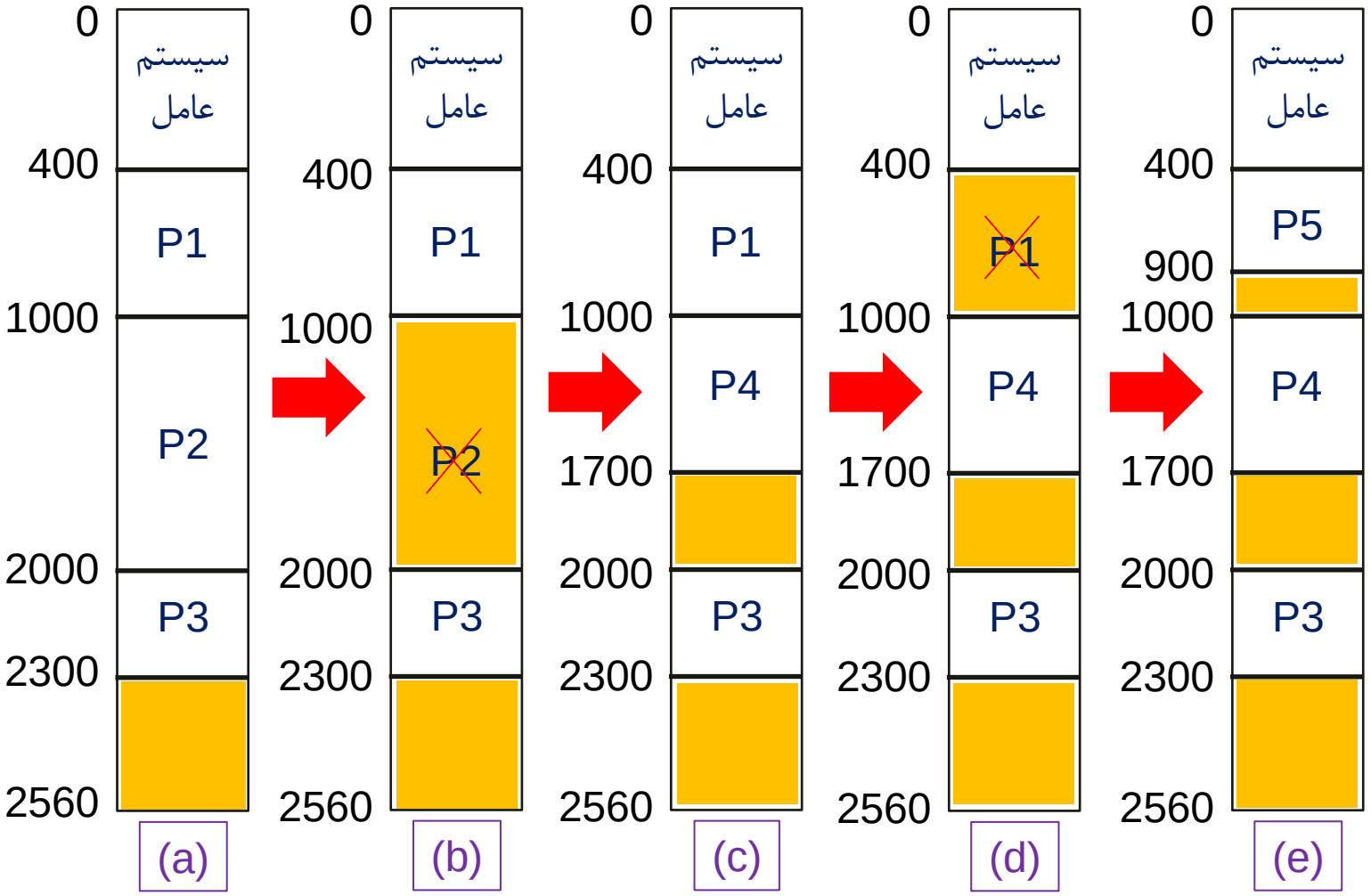
- در این تکنیک، فرآیندها به ترتیب مناسب، پشت سر هم در حافظه بار می شوند. اگر حافظه لازم برای فرآیند نباشد، سیستم عامل صبر می کند تا حافظه به اندازه مورد نیاز خالی شود.



صف فرآیندها		
فرآیند	حافظه مورد نیاز	زمان اجرا
P1	600K	10
P2	1000K	5
P3	300K	20
P4	700K	8
P5	500K	15

تخصیص چند بخشی همجوار

مثال:



مدیریت فضاهای آزاد

- مدیر حافظه در هر لحظه باید اطلاع داشته باشد که کدام قسمت حافظه پُر و کدام قسمت حافظه خالی می باشد. برای این منظور از دو روش زیر استفاده می کند :

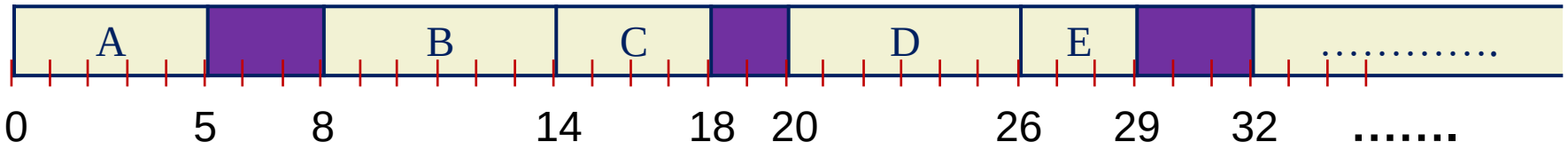
✓ روش نگاشت بیتی

- در این روش حافظه به قسمت های مساوی به نام واحد تخصیص (Allocation Unit) تقسیم می شود. متناظر با هر واحد تخصیص، یک بیت وجود خواهد داشت که اگر آن واحد پُر باشد بیت متناظر آن 1 و اگر آن واحد خالی باشد بیت متناظر آن 0 خواهد شد.

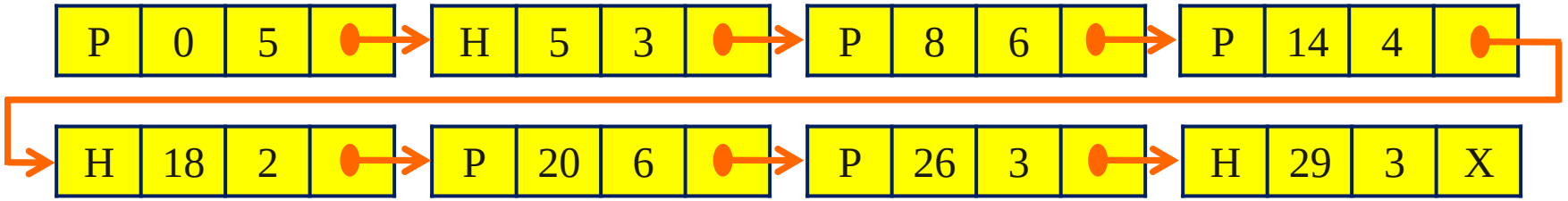
✓ روش لیست پیوندی

- در این روش هر فضای آزاد یا هر فرآیند توسط یک گره لیست پیوندی به نمایش در خواهد آمد.

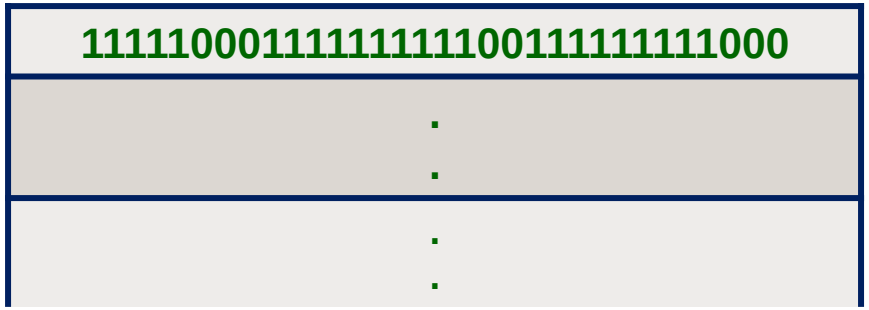
مثال: حافظه زیر را در نظر بگیرید.



روش لیست پیوندی :



روش نگاشت بیتی :



روش های تخصیص حافظه

- در روش بخش بندی پویا مشاهده کردیم که در هر لحظه لیستی از فضاهای خالی وجود دارد تا فرآیندها را بتوان در مناسب ترین فضای خالی قرار داد.

سوال: اگر چند فضای خالی مناسب وجود داشته باشد، سیستم عامل فرآیند را در کدام فضا قرار می دهد ؟

پاسخ: به این عملیات، تخصیص حافظه (به صورت همجوار) می گویند و به روش های زیر اجرا می گردد:

۱. اولین مناسب (First Fit)

۲. مناسب بعدی (Next Fit)

۳. بهترین مناسب (Best Fit)

۴. بدترین مناسب (Worst Fit)

۵. سریعترین مناسب (Quick Fit)

۶. الگوریتم رفاقتی (Buddy)

اولین مناسب (First Fit)

- در این روش هنگام ورود یک فرآیند، سیستم عامل در لیست فضاهای خالی، اولین فضای آزاد را که برای فرآیند جا داشته باشد، به فرآیند اختصاص می دهد.
- شروع جستجو همیشه از ابتدای لیست فضاهای آزاد می باشد.
- تراکم اشغال حافظه در ابتدای حافظه بیشتر خواهد بود.
- بلاک های کوچکتر، بیشتر از ابتدای حافظه پُر می شوند.

مناسب بعدی (Next Fit)

- مانند روش اولین مناسب جستجو می کند.
- تفاوت این روش با اولین مناسب در این است که روش مناسب بعدی بر خلاف روش اولین مناسب، هر مرتبه جستجو را از ابتدای لیست شروع نمی کند، بلکه جستجو از مکان انتهای جستجوی قبلی آغاز می شود.
- مناسب بعدی نسبت به روش قبلی، یکنواختی بهتری در توزیع برنامه ها در حافظه دارد.
- زمان کمتری برای جستجوی فضای خالی صرف می شود.
- حافظه را به مقدار زیادی تکه تکه می کند.

بهترین مناسب (Best Fit)

- در این روش کل لیست فضاهای آزاد جستجو می شود تا کوچکترین فضای مناسب با اندازه فرآیند پیدا شده و به فرآیند اختصاص یابد.
- در این روش فضاهای کوچکتر، سریع تر اختصاص یافته و سریع تر تکه تکه می شوند.
- در این روش فضاهای بزرگ برای فرآیند های بزرگ حفظ می شوند.
- به دلیل جستجو در تمام لیست، این روش قدری زمان بیشتری نیاز دارد.

بدترین مناسب (Worst Fit)

- در این روش کل لیست فضاهای آزاد جستجو می شود تا بزرگترین فضای موجود پیدا شده و به فرآیند اختصاص یابد.

- تفاوت آن با روش بهترین مناسب در این است که احتمال بیشتری وجود دارد تا تکه های فضای خالی ایجاد شده در این روش برای دیگر فرآیندها قابل استفاده باشد.

- مشکل این روش این است که به دلیل تکه کردن فضاهای بزرگ، ممکن است فرآیندهای بزرگ امکان قرار گرفتن در حافظه را نداشته باشند.

سریع ترین مناسب (Quick Fit)

- در این روش برای فرآیندهای هم اندازه لیست جداگانه تهیه شده و یک جدول N خانه ای وجود دارد که هر خانه به فضاهای خالی هم اندازه ای اشاره دارد.
  - ✓ مثلاً خانه اول فضاهای خالی 4 KB، خانه دوم فضاهای خالی 8 KB و خانه سوم ... اِلی آخر.

- برای هر دسته فرآیند فقط در لیست های مناسب به آنها جستجو می شود.

- به دلیل دسته بندی فضاها و فرآیندها، جستجو سریعتر انجام می شود.

- مشکل این روش اینست که در زمان انتهای کار یک فرآیند، باید اطلاعات فضای خالی شده آن فرآیند را به لیست مناسب اضافه کند که قدری زمان بَر خواهد بود.

الگوریتم رفاقتی(Buddy)

- در این روش بخش های آزاد حافظه به قطعاتی با اندازه های توان ۲ تقسیم می شوند و برای هر گروه لیست جداگانه ای در نظر گرفته می شود.
- هر فرآیند در یکی از فضاهایی قرار می گیرد که از نظر اندازه برایش مناسب تر است.
- پس از اتمام کار هر فرآیند فضاهای آزاد مجاور هم در صورت امکان با هم ترکیب شده تا فضای بزرگتری ایجاد شود.
- سرعت تخصیص حافظه بالا می باشد.

مشکلات مدیریت حافظه

۱) تکه تکه شدن داخلی (Internal Fragmentation)

- وقتی یک فضای خالی از حافظه به یک فرآیند اختصاص پیدا می کند که اندازه آن از اندازه فرآیند بیشتر است، به فضای حافظه ای که در این تکه فضا غیر قابل استفاده خواهد ماند، تکه تکه شدن داخلی گفته می شود.

- این مشکل بیشتر در بخش بندی ایستا و روش بهترین مناسب رخ می دهد.

- به این مشکل پارگی داخلی نیز گفته می شود.

مشکلات مدیریت حافظه

۲) تکه تکه شدن خارجی (External Fragmentation)

- وقتی در حافظه فضاهای آزاد غیر همجوار وجود داشته باشند که هیچ کدام به تنهایی برای فرآیندها مناسب نبوده ولی مجموع آنها به اندازه فرآیندها باشد، در این وضعیت است که تکه تکه شدن خارجی رخ داده است.

- این مشکل بیشتر در بخش بندی ایستا و روش اولین مناسب رخ می دهد.

- به این مشکل پارگی خارجی نیز گفته می شود.

مشکلات مدیریت حافظه

۳) پراکندگی برنامه ها (Sparseness)

- ✓ پراکندگی استاتیک (ایستا)
- ✓ پراکندگی دینامیک (پویا)

۴) استفاده مشترک از کُد و داده ها

سؤال ؟