دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)
دانشکدهٔ مهندسی کامپیوتر و فنآوری اطلاعات

اصول و مفاهیم سیستم عامل

ترم بهمن ۹۷

فصل چهارم: مدیریت حافظهٔ اصلی

نستوه طاهری جوان

nastooh@aut.ac.ir

سلسله مراتب حافظه

- برنامه نویسان تمایل دارند یک حافظه با وسعت بی نهایت، سرعت بسیار بالا و غیر فرار (پایدار) در اختیار داشته باشند، اما در عمل چنین چیزی در دسترس نیست، در عوض اکثر سیستم ها از یک حافظهٔ سلسله مراتبی استفاده می کنند.

یک نمونه رایج از سلسله مراتب حافظه

nastooh@aut.ac.ir

مدیریت حافظه

- یکی از مهمترین وظایف سیستم عامل مدیریت حافظهٔ اصلی است. منظور از حافظهٔ اصلی، حافظه ای است که پردازنده برای دستیابی به دستورالعملها و داده ها مستقیم به آن رجوع می کند. (مانند RAM)
- برخی از مسائل مطرح در مدیریت حافظهٔ اصلی
  - در آن واحد چند برنامه در حافظه باشد؟
  - سهم بندی برنامه ها و کاربران از حافظه چگونه است؟
  - کدام قسمت ها در اختیار کدام برنامه ها است؟
  - مدیریت بخش های خالی حافظه.
  - کنترل ورود و خروج برنامه ها به/از حافظه.
  - و ...

nastooh@aut.ac.ir

پیوند آدرس

- ✓ آدرس ها در برنامهٔ نوشته شده توسط برنامه نویس، به صورت سمبلیک هستند. مثلا:

```
int  i;
```

  - در این حالت i یک نام سمبلیک برای دو بایت خاص از حافظه است.
  - **سوال اصلی**: این دو بایتی که در اختیار متغیر i است، واقعا کجای حافظه و در کدام آدرس قرار دارد؟
  - اگر این برنامه دوباره اجرا شود، باز هم متغیر i در همان مکان قرار می گیرد؟
  - اگر این برنامه بر روی یک ماشین دیگر (با مشخصات و اندازهٔ حافظهٔ متفاوت) اجرا شود، تکلیف متغیر i چیست؟

nastooh@aut.ac.ir

پیوند آدرس

- روال کلی کار:
  1. برنامه نویس از آدرس های سمبلیک استفاده می کند.
  2. مترجم (کامپایلر) آدرس های سمبولیک را به آدرس های قابل جابجایی تبدیل می کند.
  3. لودر باید آدرس های قابل جابجایی را (هنگام اجرا) به آدرس های مطلق تبدیل کند.

nastooh@aut.ac.ir

پیوند آدرس

- ✓ زمان تبدیل نهایی آدرس به آدرس مطلق
  - زمان کامپایل
    - اگر در زمان ترجمه مشخص باشد یک برنامه در چه بخشی از حافظه قرار می گیرد، در این صورت کد مطلق تولید می شود.
    - این برنامه ها همیشه باید در یک جای ثابت بارگذاری شوند.
    - برنامه های از نوع COM در سیستم عامل DOS از این نوع هستند.
  - زمان بارگذاری
    - مصطلح ترین حالت.
    - کامپایلر کد قابل حمل با آدرس های نسبی تولید می کند.
    - برنامه های EXE در سیستم عامل DOS از این نوعند.
  - زمان اجرا
    - حالت بسیار خاص، فرآیند بتواند در حین اجرا مکانش را تغیر دهد!

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ تک برنامگی
  - تقسیم حافظه به صورت ساده و به سه بخش:
    - یک بخش ثابت در اختیار سیستم عامل.
    - یک بخش در اختیار برنامهٔ کاربردی.
    - مابقی حافظه: بلا استفاده.
  - نکات:
    - یک برنامه کل حافظه، به جز بخش سیستم عامل را می تواند در اختیار بگیرد.
    - عدم استفادهٔ بهینه از حافظه.
    - محدود بودن اندازهٔ برنامه ها به اندازهٔ حافظه.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ تک برنامگی (ادامه)
  - ○ تکنیک جایگذاشت (Overlay)
    - هدف: اجرای برنامه های بزرگتر از حافظه.
    - راهکار: هر زمان که داده یا کدی از برنامه مورد نیاز است، به حافظه آورده شود.
      - ➢ در واقع تا جایی که حافظه گنجایش دارد از فرآیند مورد نظر به حافظه منتقل می شود، اما هنگامی که بخش دیگری نیاز باشد، آن بخش به حافظه آورده می شود.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ چند برنامگی
  - در آن واحد چندین برنامه در حافظه قرار دارند.
  - عملا به دو صورت کلی انجام می شود:

1. بخش بندی ایستا
   a) بخش های مساوی
   b) بخش های نامساوی
      i. ***صف جداگانه برای هر بخش***
      ii. ***صف یکتا برای هر بخش***

2. بخش بندی پویا

nastooh@aut.ac.ir

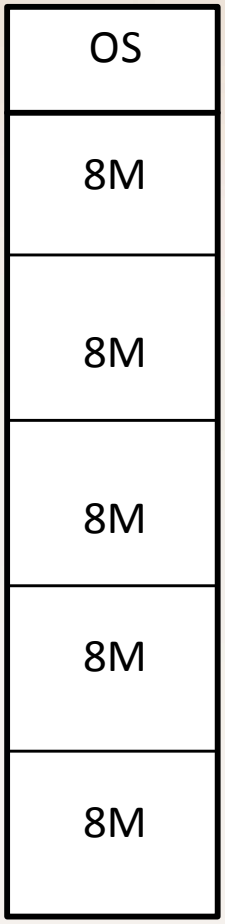
مقدمات تخصیص حافظه

- ✓ چند برنامگی (ادامه)
  - ○ بخش بندی ایستا
    - سیستم عامل یک بخش از حافظه را در اختیار می گیرد.
    - مابقی حافظه در قالب بخش هایی با اندازهٔ **ثابت** در اختیار برنامه ها قرار می گرد.
    - هر برنامه که اندازهٔ آن مساوی یا کمتر از آن بخش باشد، می تواند به آن بخش منتقل شود.
    - وقتی یک فرآیند به یک بخش وارد شد، مابقی فضای بخش غیر قابل استفاده است. به این اتفاق تکه تکه شدن داخلی (Internal Fragmentation) گویند.
    - می توان اندازهٔ بخش ها را نامساوی نیز در نظر گرفت (کماکان ثابت). به این ترتیب هر فرآیند می تواند به کوچکترین بخشی که در آن جا می شود، وارد شود.

nastooh@aut.ac.ir

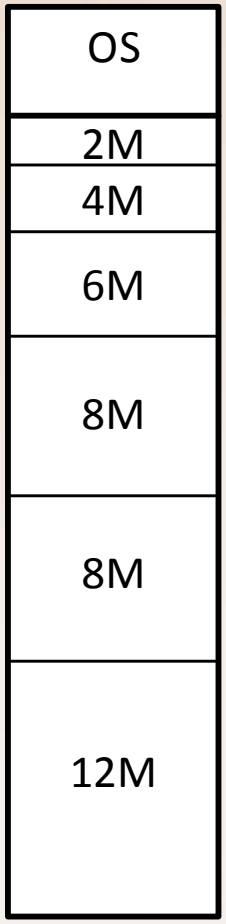
مقدمات تخصیص حافظه

- ✓ چند برنامگی
  - بخش بندی ایستا



قسمت های مساوی

مثالی از بخش پذیری ایستای یک حافظهٔ ۴۰ مگا بایتی



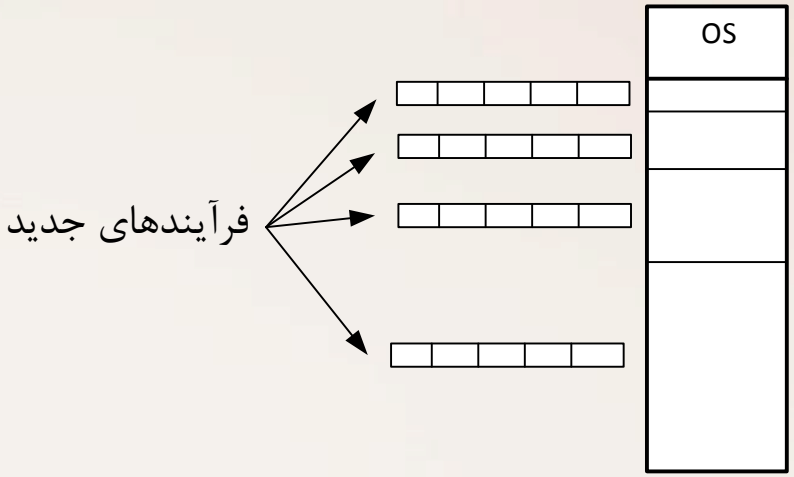
قسمت های نامساوی

nastooh@aut.ac.ir

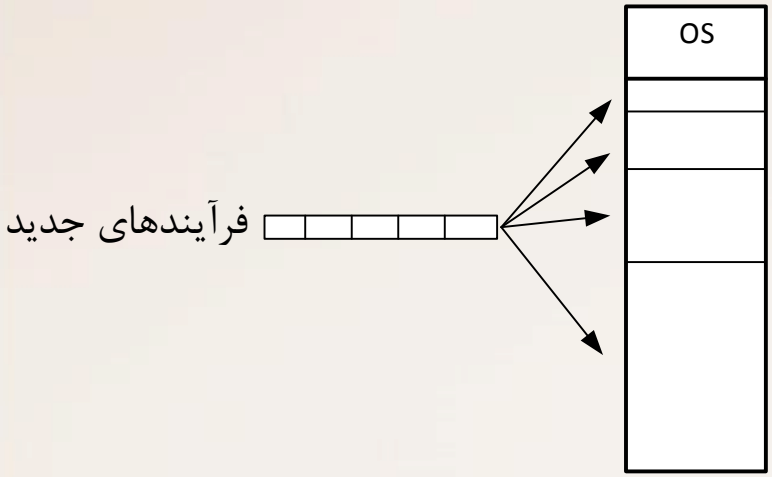
مقدمات تخصیص حافظه

- چند برنامگی
  - بخش بندی ایستا
    - بخش های نامساوی

➢ دو راه برای تخصیص فرآیندها به بخش ها وجود دارد. یکی استفاده از یک صف واحد و دیگری استفاده از صف های جدا گانه برای بخش ها.



عیب: ممکن است یک صف با اندازهٔ کوچک با ازدحام مواجه شده باشد اما صف های با اندازهٔ بزرگتر خالی باشند!!



عیب: ممکن است یک فرآیند کوچک، چون قطعهٔ مناسب پر است، در یک قطعهٔ بزرگ جا شود، در صورتی که اگر قدری منتظر می ماند، ممکن بود قطعه کوچک آزاد شود.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ چند برنامگی
  - ایرادهای بخش بندی ایستا
    - تعداد فرآیندهای فعال، محدود به تعداد بخش های تعریف شده است.
    - کارهای کوچک، موجب ناکارآمدی و تکه تکه شدن داخلی می شوند.
  - راه کار:
    - استفاده از بخش بندی پویا

➢ ایدهٔ کلی: بخش های استفاده شده، دارای طول متغیر باشند و تعداد آنها نیز ثابت نباشد.

« به عبارت دیگر، هر فرآیند دقیقا به اندازهٔ مورد نیاز حافظه در اختیار بگیرد.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ چند برنامگی
  - ○ بخش بندی پویا
    - در ابتدا کل حافظه همانند یک بلوک آزاد در نظر گرفته می شود.
    - وقتی فرآیندی درخواست حافظه می کند، به همان اندازه حافظه به آن تخصیص می دهیم و مابقی حافظه برای درخواست های بعدی آزاد می ماند.
    - به همین ترتیب فرآیندهای بعدی با درخواست های جدید وارد می شوند.
    - هر گاه فرآیندی به پایان رسید، حافظهٔ در اختیار آن آزاد محسوب می شود.
    - نکته مهم: این روش به خوبی آغاز می شود، اما در نهایت حفره های کوچک بسیاری در حافظه پدید می آید. این پدیده را تکه تکه شدن خارجی (External Fragmentation) گویند.
      - ➢ در واقع وقتی یک فرآیند حافظه را ترک می کند، یک بخش آزاد بر جا می گذارد. حال احتمال اینکه فرآیند جایگزین دقیقا به همان اندازه باشد بسیار بسیار ضعیف است. پس یک فرآیند کوچک تر در آنجا قرار می گیرد و مابقی فضایی که آزاد می ماند، احتمالاً آنقدر کوچک است که دیگر هیچ فرآیندی در آن جا نمی شود! ☹

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ تفاوت تکه تکه شدن داخلی و خارجی!
  - تکه تکه شدن داخلی زمانی رخ می دهد که حافظه از قبل مرزبندی شده باشد و در هر قسمت فقط بتوان یک فرآیند را قرار داد. به این ترتیب مقداری از فضای انتهای قسمت غیر قابل استفاده است.
  - اما اگر از قبل مرزبندی مشخصی نداشته باشیم، به دلیل رفت و آمدهای پیاپی فرآیندها، فضاهای خالی و کوچک بین فرآیندها عموماً غیر قابل استفاده هستند. به این مشکل تکه تکه شدن خارجی گویند.
    - البته می توان با استفاده از تکنیک پر هزینهٔ Compaction این مشکل را برطرف کرد و حفره ها را برای استفاده های آینده همجوار کرد. هزینهٔ این عمل بسیار بالاست، زیرا علاوه بر جابجایی فرآیندها، سیستم عامل باید پیوند آدرسهای فرآیندها را به درستی و با دقت تغییر دهد.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ روش های نگهداری وضعیت حافظه
  - ○ سیستم عامل باید از قسمت های آزاد و اشغال حافظه مطلع باشد.
    - روش اول: روش بیت مپ
      - ➢ حافظه به واحدهای کوچکی تقسیم می شود. (کوچکترین واحد قابل تخصیص)
      - ➢ سپس متناظر با هر واحد، یک بیت در یک دنبالهٔ بیتی در نظر میگیریم.
      - ➢ حال اگر واحدی آزاد باشد، بیت متناظر با آن صفر و اگر اشغال باشد یک می شود.
      - ➢ برای تخصیص nواحد به یک فرآیند جدید، باید به دنبال n صفر پشت سر هم بود.
    - روش دوم: روش لیست پیوندی
      - ➢ هر گرهٔ لیست پیوندی دو حالت دارد، یا یک حفره است، یا یک فرآیند.
      - ➢ در هر گرهٔ این لیست باید اطلاعاتی از قبیل: آدرس شروع قطعه، طول آن، وضعیت پر و خالی بودن آن و آدرس شروع قطعه بعدی درج شده باشد.
      - ➢ این روش قدری وقت گیر است. مثلا هنگام ترک یک فرآیند، باید همسایه های آن فرآیند جهت ترکیب حفره های احتمالی پیدا شوند.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- ✓ روش های تخصیص حافظه به فرآیندها
  - در این بخش فرض می کنیم پس از مدتی تعدادی فرآیند در حافظه قرار دارند و بین آنها تعدادی حفرهٔ آزاد با اندازه های مختلف وجود دارند. حال سوال این است، فرآیند تازه وارد در کدام قسمت قرار داده شود؟
  - روش First Fit
    - در این روش، سیستم عامل حافظه را از ابتدا جستجو می کند، اولین فضای حافظه ای که فرآیند در آن بگنجد انتخاب می شود.
    - تراکم فرآیندها معمولا در ابتدای حافظه بیشتر است.
  - روش Next Fit
    - مشابه روش First Fit، با این تفاوت که جستجو برای یافتن اولین محل مناسب، از محل آخرین تخصیص شروع می شود. (نه هواره از ابتدای حافظه)

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- روش Best Fit
  - در این روش کل فضای حافظه باید جستجو شود. سپس کوچکترین حفره ای که بتواند فرآیند را در بر بگیرد، انتخاب می شود.
    - در واقع منطق و ایدهٔ این روش این است که حفره های بزرگ را نباید تقسیم کرد. زیرا بعدها ممکن است به آن ها نیاز داشته باشیم.
- روش Worst Fit
  - در این روش نیز، کل حافظه جستجو می شود، تا همواره بزرگترین حفره انتخاب شود!
    - ایده و منطق این روش این است که بعد از تخصیص بزرگ ترین حفره به یک فرآیند، فضای باقیمانده احتمالاً آنقدر بزرگ هست که یک فرآیند دیگر را در خود جای دهد. اما در روش Best Fit احتمالا حفرهٔ ایجاد شده آنقدر کوچک است که دیگر به درد نمی خورد.

نکته: در روش Best Fit حافظه پر از حفره های کوچک می شود که به هیچ کاری نمی آیند. در مقابل در روش Worst Fit فرآیندهای بزرگ ممکن است با گرسنگی مواجه شوند.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- روش Quick Fit
  - در این روش لیست های جداگانه ای برای تخصیص حافظه به فرآیندهای با اندازه های متداول تهیه می شود.
  - مثلا در یک جدول درایهٔ اول یک اشاره گر است به ابتدای یک لیست از حفره های 4k و درایهٔ بعد یک اشاره گر به ابتدای یک لیست از حفره های 8k و الی آخر.
- روش Buddy (رفاقتی)
  - در این روش اندازهٔ بخش های حافظه همگی توانی از ۲ هستند. یعنی کل حافظه یک بلوک با اندازهٔ 2^u در نظر گرفته می شود.
  - حال اگر درخواستی با اندازهٔ s مطرح شود که $2^{u-1}<s<2^u$ باشد، تمام بلوک تخصیص می یابد. در غیر اینصورت این بلوک به دو رفیق با اندازه های 2^{u-1} تقسیم می شود.
  - اگر $2^{u-2}<s<2^{u-1}$ باشد، یکی از این دو رفیق به فرآیند تخصیص می یابد. در غیر این صورت یکی از رفقا به دو رفیق دیگر تقسیم می شود.
  - روش رفاقتی مشکل تکه تکه داخلی دارد.

nastooh@aut.ac.ir

مقدمات تخصیص حافظه

- مثال از روش رفاقتی

فضای یک مگابایتی آزاد	1M					
درخواست A=100K	**A**	128K	256K	512K		
درخواست B=200K	**A**	128K	**B**	512K		
درخواست C=60K	**A**	**C**	64K	**B**	512K	
درخواست D=200K	**A**	**C**	64K	**B**	**D**	256K
آزاد سازی B	**A**	**C**	64K	256K	**D**	256K
آزادسازی A	128K	**C**	64K	256K	**D**	256K
درخواست E=75K	**E**	**C**	64K	256K	**D**	256K
آزادسازی C	**E**	128K		256K	**D**	256K
آزادسازی E	512K				**D**	256K
آزادسازی D	1M					

nastooh@aut.ac.ir

تخصیص غیر همجوار حافظه

✓ روش قطعه بندی

- در این روش فضای آدرس منطقی فرآیند به چندین قطعه تقسیم می شود.
  - در واقع فضای آدرس هر فرایند متشکل از اجتماعی از قطعات است.
  - هیچ لزومی ندارد اندازهٔ قطعات برابر باشند.
  - معمولا داده های مرتبط با هم در یک قطعه قرار می گیرند.
    - مانند زیربرنامه ها یا داده ها مانند یک آرایهٔ بزرگ!
  - برای اجرای یک فرآیند همهٔ قطعات باید به حافظه آورده شوند اما این قطعات لزوماً مجاور یکدیگر قرار نمی گیرند.
  - هر فرآیند باید یک جدول با عنوان جدول قطعه داشته باشد.
    - به ازای هر قطعه یک درایه در این جدول وجود دارد که مشخص می کند هر قطعه در کجای حافظه قرار گرفته است.
  - در این روش کماکان احتمال تکه تکه شدن خارجی وجود دارد.

nastooh@aut.ac.ir

تخصیص غیر همجوار حافظه

✓ قطعه بندی (ادامه)

○ فرآیند تبدیل آدرس در قطعه بندی

- در هر درایهٔ جدول قطعه، آدرس شروع قطعه در حافظهٔ اصلی و طول قطعه ذخیره می شود. به طول قطعه معمولاً limit و به آدرس شروع قطعه معمولاً Base گفته می شود.
- آدرس های منطقی به صورت زیر هستند.

m بیت	n بیت
آفست	شماره قطعه

- در این حالت از n بیت مربوط به شماره قطعه، برای ارجاع به جدول قطعه استفاده می شود و آدرس فیزیکیِ شروع این قطعه پس از استخراج از جدول با آفست **جمع جبری** می شود.
- آفستِ آدرس منطقی باید کوچکتر از limit باشد، بنابراین ابتدا آفست با limit مقایسه می شود، اگر آفست از limit بزرگتر بود، وقفهٔ خطای دسترسی غیرمجاز صادر می شود.

nastooh@aut.ac.ir

تخصیص غیر همجوار حافظه

- روش صفحه بندی
  - حافظه به بخش هایی با اندازهٔ مساوی با نام Frame تقسیم می شود.
  - برنامه ها نیز به بخش هایی با اندازه های مساوی با قاب ها به نام Page تقسیم می شوند.
  - هنگام اجرای یک برنامه باید تمام صفحات آن، به داخل قاب های خالی آورده شود.
    - در این حالت نیازی نیست صفحات در قاب های مجاور لود شوند.
  - این روش از دید برنامه نویس پنهان است.
  - در این حالت کوچکترین واحد تخصیص حافظه، یک صفحه است.
    - معمولا اندازهٔ صفحات توانی از ۲ در نظر گرفته می شود.
  - هر فرآیند نیاز به یک جدول صفحه دارد که نشان می دهد کدام صفحه در کدام قاب حافظه قرار گرفته است.

nastooh@aut.ac.ir

تخصیص غیر همجوار حافظه

- ✓ روش صفحه بندی (ادامه)
  - نحوهٔ تبدیل آدرس
    - آدرس های منطقی از دو بخش تشکیل شده است، شمارهٔ صفحه و آفست.
    - با استفاده از شمارهٔ صفحه، به سراغ درایهٔ جدول صفحه می رویم و **شمارهٔ قاب** مورد نظر در حافظه را استخراج می کنیم.
    - سپس شمارهٔ قاب را به جای شمارهٔ صفحه در آدرس منطقی قرار می دهیم. (قسمت آفست بدون تغییر باقی می ماند.)
    - لزوماً شمارهٔ صفحه با شمارهٔ قاب هم اندازه نیست. (معمولا شماره قاب بلندتر از شمارهٔ صفحه است.)

nastooh@aut.ac.ir

تخصیص غیر همجوار حافظه

- ✓ روش صفحه بندی (ادامه)
  - روش صفحه بندی تکه تکه شدن خارجی ندارد، اما قدری تکه تکه شدن داخلی دارد. (به ازای آخرین صفحهٔ هر فرآیند.)
  - نگهداری جدول صفحه در حافظهٔ اصلی
    - در این حالت برای هر بار دسترسی به حافظه باید ۲ بار به آن مراجعه کنیم. یک بار برای به دست آوردن شماره قاب، و یک بار برای دسترسی به دادهٔ مورد نظر.
  - دو عیب ذاتی تکنیک صفحه بندی
    - یک، اندازهٔ بزرگ جداول صفحه در عمل
      - استفاده از تکنیک صفحات چند سطحی (مزایا و معایب)
      - استفاده از جداول صفحه معکوس (مزایا و معایب)
    - دو، نیاز به حداقل دو دسترسی به حافظه (و در نتیجه کاهش سرعت)
      - استفاده از تکنیک TLB بر اساس اصل مراجعات محلی

nastooh@aut.ac.ir

تمرین ها

- تمرین ۱: دربارهٔ ثبات PTBR تحقیق کنید.
  - مرجع [1]، صفحه ۳۷۲.

- تمرین ۲: الف) دربارهٔ تکنیک TLB جهت افزایش سرعت صفحه بندی تحقیق کنید.
  - مرجع [2]، بخش 3-3-3
  - مرجع [1]، بخش 2-5-8

ب) فرض کنید در سیستمی زمان دسترسی به حافظه ۶۰ نانو ثانیه و زمان دسترسی به TLB برابر ۵ نانو ثانیه است. اگر احتمال وجود شمارهٔ صفحه در TLB برابر ۸۰٪ باشد، **میانگین** زمان دسترسی به حافظه دقیقا چقدر است؟

nastooh@aut.ac.ir

تمرین ها

- ✓ تمرین ۳: دربارهٔ جداول صفحهٔ معکوس (Inverted Page Tables) تحقیق کنید. مزایا و معایب آن را بررسی کنید.
  - مرجع [3]، صفحه ۳۴۹.
- ✓ تمرین ۴: دربارهٔ جداول صفحهٔ درهم (Hashed) تحقیق کنید.
  - مرجع [1]، بخش 2-6-8
- ✓ تمرین ۵: دربارهٔ قطعه بندی همراه با صفحه بندی تحقیق کنید.
  - مرجع [4]، بخش 2-6-4
  - مرجع [3]، صفحه ۳۵۷

nastooh@aut.ac.ir

تمرین ها

- تمرین ۶: الف) دربارهٔ صفحه بندی چند سطحی تحقیق کنید. مزیت عمدهٔ آن چیست؟
  - مرجع [1]، بخش 8-6-1
  - مرجع [4]، شکل 4-10

ب) سیستمی از جداول صفحهٔ دو سطحی استفاده می کند و دارای آدرس های منطقی ۳۲ بیتی است. اولین هشت بیت آدرس به عنوان ایندکس به جدول سطح اول اشاره دارد و ۱۰ بیت بعدی عناصر جداول سطح دوم را مشخص می کند. به سوالات زیر پاسخ دهید:

1. صفحات در این سیستم چند بایتی هستند؟
2. فضای آدرس دهی منطقی چند صفحه را شامل می شود؟
3. جداول سطح اول و سطح دوم **هر کدام** چند درایه دارند؟

nastooh@aut.ac.ir

تمرین ها

- ✓ تمرین ۷: سیستمی از قطعه بندی ساده استفاده می کند، آدرس فیزیکی هر یک از آدرس های منطقی داده شده را پیدا کنید.

Segment	Base	Limit
0	330	124
1	876	211
2	111	99
3	498	302

- ○ آدرس ها:

0	110
2	80
1	231
3	210
2	110

nastooh@aut.ac.ir

تمرین ها

- تمرین ۸: چرا سیستم عامل های موبایل مانند iOS و اندروید از تکنیک مبادله (Swapping) استفاده نمی کنند؟ راهکار جایگزین آنها چیست؟
  - منبع [1]، بخش 8-2-2

نمونه تست های کنکور ارشد

✓ ارشد، دولتی، ۷۹

- در زیر بلوک های خالی حافظه به ترتیب از چپ به راست نشان داده شده اند، اگر درخواست های جدیدی برای چهار بلوک به اندازه های 20k و 30k و 20k و 35k وارد شوند و سیستم از روش Next Fit استفاده کند، وضعیت حافظه بعد از این تخصیص ها کدام است؟

A)20k, 25k, 15k, 15k, 60k, 40k

B)5k, 25k, 25k, 20k, 40k, 40k

C) 20k, 25k, 15k, 30k, 25k, 40k

D)10k, 5k, 15k, 60k, 5k

✓ ارشد، دولتی، ۸۰

- یک فضای آدرس منطقی صفحه بندی متشکل از ۳۲ صفحهٔ ۲ کیلوبایتی را که به یک فضای آدرس یک مگابایتی نگاشت شده است در نظر بگیرید. هر مدخل جدول صفحه باید چند بیتی باشد؟

الف) ۱۱ ب) ۹ ج) ۵ د) ۴

- راهنمایی، فقط کافیست تعداد قاب های حافظه اصلی را پیدا کنید، سپس اندازهٔ هر درایهٔ جدول صفحه بر اساس آن مشخص می شود.

nastooh@aut.ac.ir

نمونه تست های کنکور ارشد

✓ ارشد، دولتی، ۸۷

- در یک سیستم صفحه بندی ساده، که جدول صفحه آن ۵۱۲ عنصر ۱۵ بیتی دارد و اندازهٔ صفحات یک کیلو بایتی است. اندازهٔ فضای آدرس فیزیکی چقدر است؟ آدرس فیزیکی چند بیتی است؟

الف) 2^{26} و ۲۶ بیتی ب) 2^{25} و ۲۵ بیتی ج) 2^{24} و ۲۴ بیتی د) 2^{16} و ۱۶ بیتی

✓ ارشد، دولتی، ۸۳

- در یک سیستم حافظهٔ صفحه بندی با یک جدول صفحه حاوی ۶۴ مدخلِ ۱۰ بیتی و صفحه های با اندازهٔ هر یک ۵۱۲ بایت، یک آدرس منطقی و یک آدرس فیزیکی هر کدام چند بیت هستند؟

الف) منطقی ۱۹، فیزیکی ۱۵ ب) منطقی ۱۵، فیزیکی ۱۹
ج) منطقی ۶، فیزیکی ۱۵ د) منطقی ۱۰، فیزیکی ۶

nastooh@aut.ac.ir

نمونه تست های کنکور ارشد

- ارشد، دولتی، ۸۶
  - در یک سیستم صفحه بندی ساده، حافظهٔ فیزیکی دارای 2^{24} بایت است و ۲۵۶ صفحه فضای آدرس منطقی را تشکیل می دهد و اندازهٔ صفحات 2^{10} بایت است. کدام یک از گزینه های زیر تعداد بیت های آدرس منطقی و اندازهٔ جدول صفحه را مشخص می کند؟

    الف) ۱۸ بیت و ۲۵۶ عضو

    ب) ۱۸ بیت و ۱۶ کیلو عضو

    ج) ۲۴ بیت و ۲۵۶ عضو

    د) ۲۴ بیت و ۱۶ کیلو عضو

- ارشد، دولتی، ۸۶
  - در یک صفحه بندی ساده که جدول صفحه آن ۶۴ عضو ۱۱ بیتی دارد (یک بیت برای متعبر/نامعتبر) و صفحات ۵۱۲ بایتی می باشند، چند بیت در آدرس فیزیکی مشخص کننده شماره قاب صفحه است؟

    الف) ۶ ب) ۹ ج) ۱۰ د) ۱۱

nastooh@aut.ac.ir

نمونه تست های کنکور ارشد

✓ ارشد، دولتی، ۸۵

- در یک سیستم صفحه بندی ساده، فضای آدرس دهی منطقی متشکل از ۳۲ صفحهٔ ۴ کیلوبایتی است. اگر هر مدخل جدول صفحه یازده بیت باشد، اندازهٔ فضای آدرسی که برنامه روی آن نگاشته شده است چقدر است؟

الف) یک مگابایت ب) ۴ مگابایت ج) ۸ مگابایت د)۱۶ مگابایت

✓ ارشد، دولتی، ۸۴

- یک فضای آدرس دهی منطقی شامل هشت صفحه است و هر صفحه حاوی ۱۰۲۴ کلمه است. این فضای آدرس دهی بر روی یک فضای آدرس دهی فیزیکی حاوی ۳۲ قاب نگاشته می شود. آدرس فیزیکی حاوی چند بیت است؟

الف) ۱۱ ب) ۱۳ ج) ۱۴ د) ۱۵

nastooh@aut.ac.ir

نمونه تست های کنکور ارشد

- ارشد، آزاد، ۸۴
  - سیستمی علاوه بر ذخیره جدول صفحه در حافظهٔ اصلی، از TLB نیز استفاده می کند، اگر زمان خواندن از حافظهٔ اصلی ۵۰ نانو ثانیه و زمان خواندن از TLB برابر ۲۰ نانو ثانیه باشد و درصد کارایی سیستم بدون استفاده از TLB نسبت به استفاده از TLB برابر ۸۰٪ باشد، آنگاه نرخ برخورد TLB چقدر است؟

الف) ۱۰٪ ب) ۲۰٪ ج) ۸۰٪ د) ۹۰٪

nastooh@aut.ac.ir

منابع

[1]. A. Silberschatz, P. B. Galvin and G. Gagne, **"Operating System Concepts,"** 9th ed., John Wiley Inc., 2013.

[2] A. S. Tanenbaum and H. Bos, "**Modern Operating Systems,**" 4rd ed., Pearson, 2014.

[3] W. Stallings, **"Operating Systems,"** 8th ed., Pearson, 2014.

[4] A. S. Tanenbaum, A. S. Woodhull**, "Operating Systems Design and Implementation,"** 3rd ed., Pearson, 2006.

[5]نستوه طاهری جوان و محسن طورانی، **"اصول و مفاهیم سیستم عامل،"** انتشارات موسسه آموزش عالی پارسه، ۱۳۸۶.

nastooh@aut.ac.ir

پایان

nastooh@aut.ac.ir