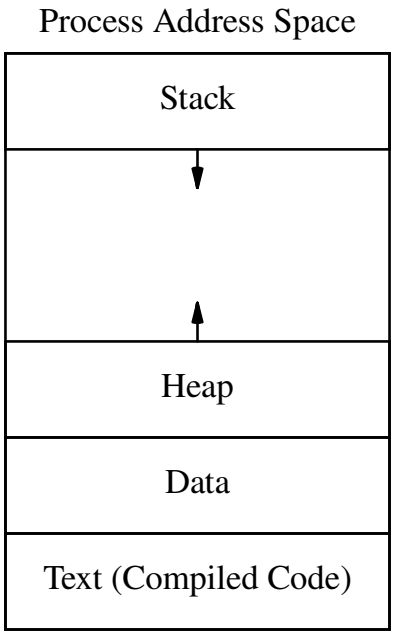
یادداشت‌های درس سیستم‌های عامل - بخش دهم

از این بخش از درس به مدیریت حافظه در سیستم عامل می‌پردازیم.

- حافظه‌ی اصلی (Main Memory) یک آرایه‌ی بزرگ از کلمه‌ها است که با آدرس مناسب می‌توان به این کلمه‌ها دسترسی داشت.
- برای اینکه یک برنامه (فایل اجرایی) اجرا شود باید به حافظه‌ی اصلی انتقال یابد.
- وقتی برنامه‌ای اجرا می‌شود، سیستم عامل قسمت‌هایی از حافظه را در اختیار پردازه قرار می‌دهد. قسمت‌های اصلی حافظه‌ی یک پردازه را در شکل زیر می‌بینید.



- قسمت Text کد ماشین قابل اجرای برنامه (که توسط کامپایلر تولید شده است) را نگه می‌دارد.
- در قسمت Data داده‌های ایستای برنامه (که در زمان کامپایل اندازه یا محتویات آنها مشخص است) قرار می‌گیرد.
- قسمت Heap مربوط به قسمتی از حافظه می‌شود که در زمان اجرا به صورت پویا به پردازه تخصیص می‌یابد.
- قسمت Stack یا پشته برای نگهداری متغیرهای محلی توابع است.

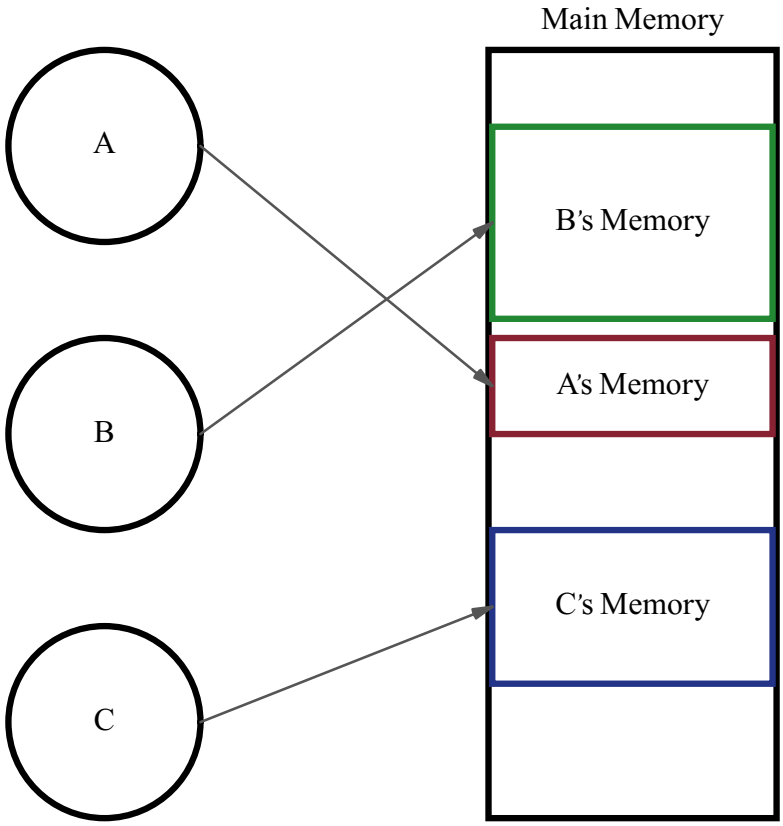
در قسمت Text کد تولید شده توسط کامپایلر قرار دارد.

- وقتی پردازه اجرا می‌شود، پردازنده دستوری که رجیستر IP (Instruction Pointer) یا PC (Program Counter) به آن اشاره می‌کند را اجرا می‌کند.
- در ادامه کد اسمبلی معادل قسمتی از کد ماشین یک پردازه نمایش داده شده است.

```
mov eax, 0x824e90
movsxd rdi, dword [rax]
test rdi, rdi
jnz 0x8094f0
mov edi, 0x818890
call qword 0x800856
```

- این دستورات به قسمت‌هایی از حافظه دسترسی دارند؛ برای نمونه دستور movsxd محتوای آدرس **0x824e90** را می‌خواند و دستور call تابعی که از آدرس **0x800856** شروع می‌شود را اجرا می‌کند.
- بنابراین در کدی که توسط کامپایلر و لینکر تولید می‌شود به آدرس‌های مشخصی از حافظه دسترسی انجام می‌شود.

- وقتی چند پردازه در یک سیستم اجرا می‌شوند، به هر پردازه قسمتی از حافظه تخصیص داده می‌شود.



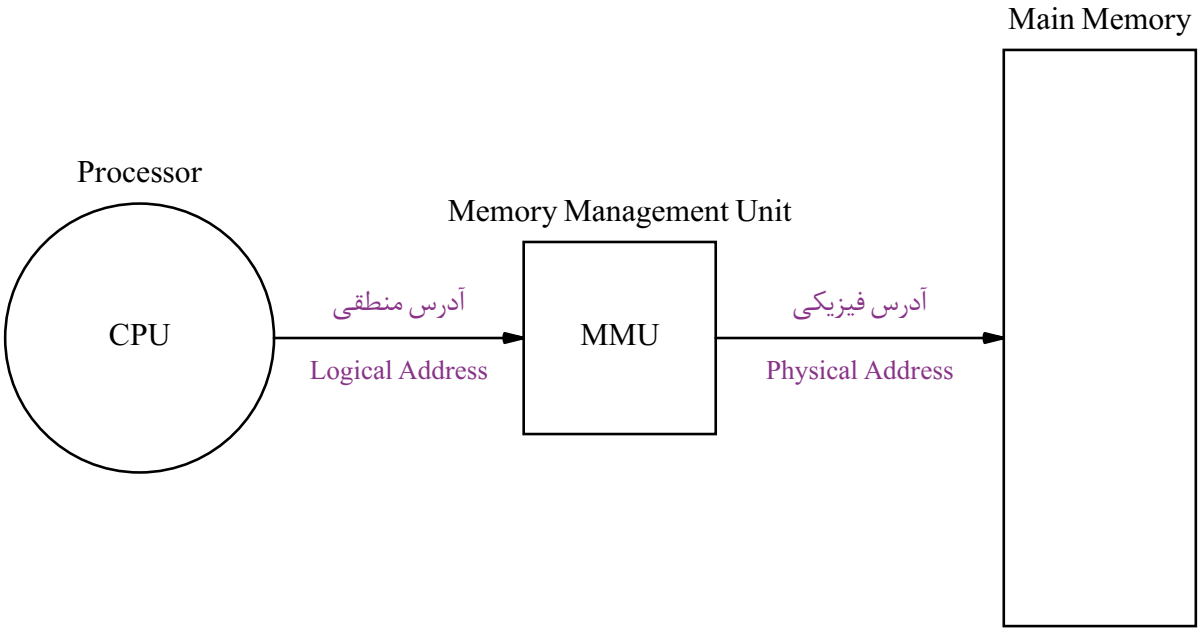
- یکی از وظایف سیستم عامل این است که تضمین کند حافظه‌ی هر پردازه مجزا است و نباید پردازه‌های دیگر به آن دسترسی داشته باشند.

اما یک مسئله‌ی مهم وجود دارد:

- اگر چند فایل اجرایی به صورت همزمان اجرا شوند، می‌توانند به آدرس‌های یکسانی از حافظه دسترسی داشته باشند.
- یا فرض کنید یک فایل اجرایی دو بار اجرا شود (برای مثال، دو پنجره‌ی مرورگر را باز کنید). این برنامه‌ها به آدرس‌های مشابهی از حافظه دسترسی دارند.
- با اجرای همزمان دو پردازه، محتویات قسمت‌هایی از حافظه به صورت ناخواسته برای هر یک از این پردازه‌ها تغییر می‌کند و این موجب می‌شود پردازه‌ها درست اجرا نشوند.

- برای حل این مشکل قطعه‌ای سخت‌افزاری به نام واحد مدیریت حافظه (Memory Management Unit یا به صورت مخفف MMU) اضافه شده است.

واحد مدیریت حافظه بین پردازنده و حافظه‌ی اصلی قرار می‌گیرد.



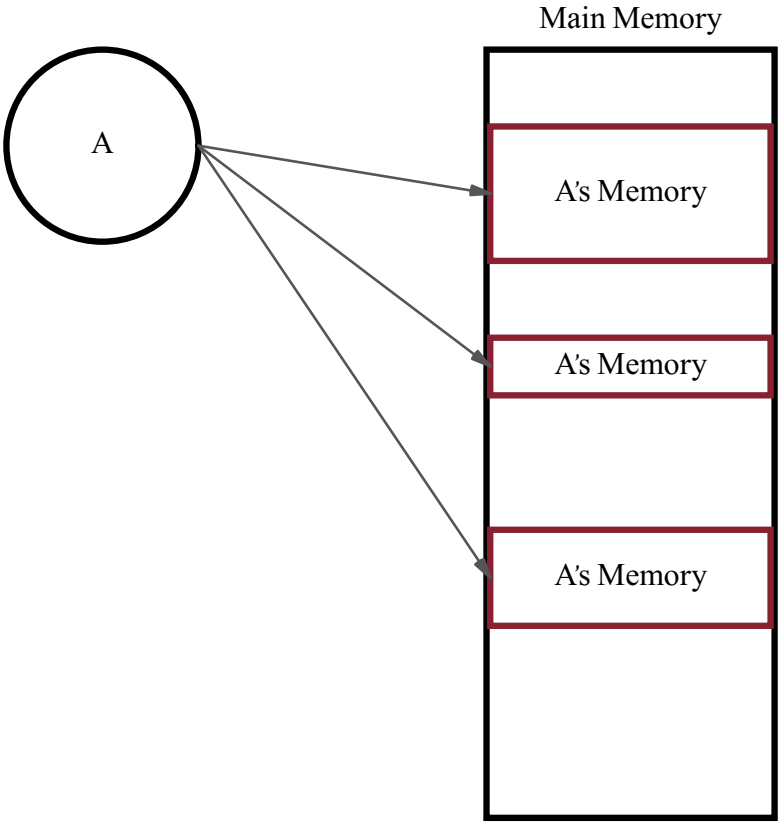
- آدرس‌هایی که در فایل اجرایی وجود دارد و توسط پردازنده اجرا می‌شود آدرس‌های منطقی (Logical Address) هستند.
- آدرس‌هایی که به حافظه‌ی اصلی فرستاده می‌شوند، آدرس فیزیکی (Physical Address) هستند.
- وظیفه‌ی اصلی واحد مدیریت حافظه نگاشت آدرس‌های منطقی به آدرس‌های فیزیکی است.
- واحد مدیریت حافظه این امکان را فراهم می‌کند که حتی اگر آدرس‌های منطقی تولید شده توسط دو پردازه کاملا یکسان باشند، این آدرس‌ها به قسمت‌های متفاوتی از حافظه‌ی اصلی نگاشت شوند.
- در پردازنده‌های امروزی MMU معمولا در خود پردازنده قرار می‌گیرد.
- در صفحه‌های بعد، شیوه‌ی کار MMU روشن‌تر می‌شود.

- به مجموعه‌ی همه‌ی آدرس‌هایی که توسط پردازنده تولید می‌شود (آدرس‌های منطقی)، فضای آدرس منطقی (Logical Address Space) گفته می‌شود.
- به مجموعه‌ی همه‌ی آدرس‌هایی که به حافظه‌ی اصلی فرستاده می‌شود (آدرس‌های فیزیکی)، فضای آدرس فیزیکی (Physical Address Space) گفته می‌شود.

- مدیریت حافظه به دو شکل انجام می‌شود:

الف) تخصیص حافظه‌ی پیوسته: به هر پردازه قسمت پیوسته‌ای از حافظه‌ی اصلی تخصیص می‌یابد.

ب) تخصیص حافظه‌ی ناپیوسته: به هر پردازه تکه‌های جدا از هم در حافظه می‌تواند تخصیص یابد.

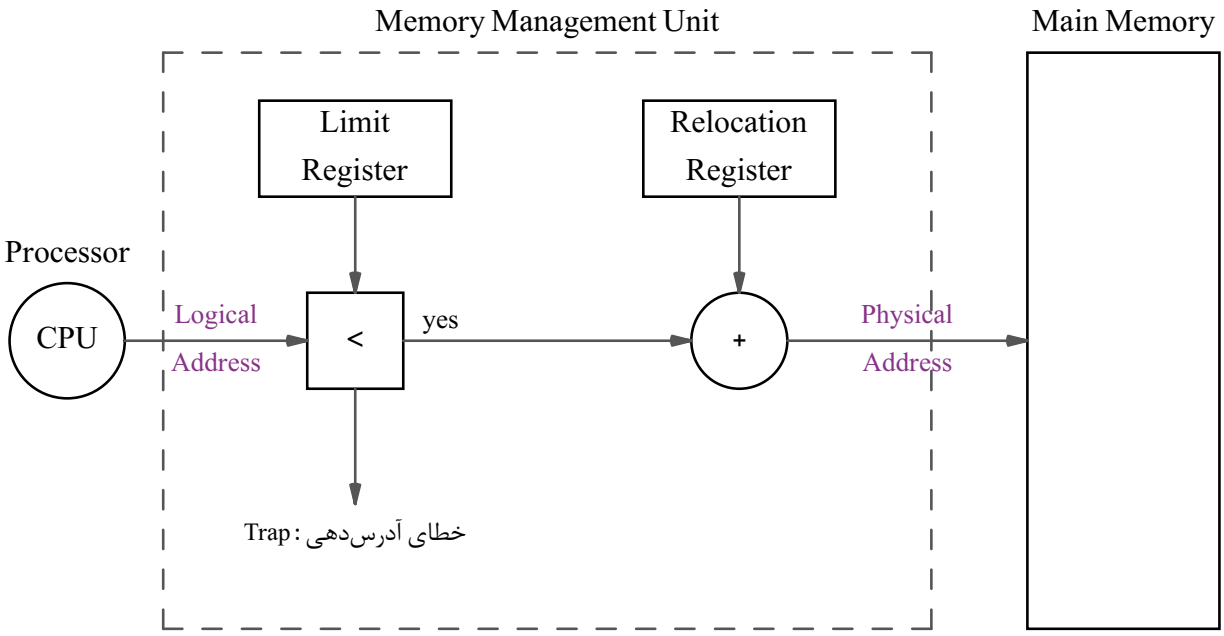


- امروزه معمولا از مدیریت حافظه‌ی ناپیوسته استفاده می‌شود.
- چون حالت پیوسته ساده‌تر است، ابتدا آن را بررسی می‌کنیم.
- در شکل زیر سه تکه از حافظه‌ی اصلی به یک پردازه تخصیص داده شده است؛ بنابراین تخصیص حافظه به صورت ناپیوسته انجام شده است.

در تخصیص پیوسته‌ی حافظه دو رجیستر برای مدیریت حافظه وجود دارند:

Relocation: شروع حافظه‌ی اختصاص یافته به پردازه را مشخص می‌کند.

Limit: اندازه‌ی حافظه‌ی تخصیص یافته به پردازه را مشخص می‌نماید.



- تخصیص حافظه به صورت شکل زیر انجام می‌شود:

ابتدا آدرس منطقی با مقدار رجیستر Limit مقایسه می‌شود؛

اگر آدرس از مقدار بزرگ‌تر یا مساوی بود یعنی آدرس غیر مجاز است و به بیرون حافظه‌ی تخصیص یافته اشاره می‌کند.

در غیر این صورت، با جمع کردن آدرس منطقی و رجیستر Relocation آدرس فیزیکی بدست می‌آید.

آدرس فیزیکی به حافظه‌ی اصلی فرستاده می‌شود.

تخصیص حافظه‌ی پیوسته - مثال

- فرض کنید در سیستم عاملی اندازه‌ی حافظه صد بایت باشد (برای سادگی در درک، از اندازه‌های بسیار کوچک در مثال‌ها استفاده می‌شود).
- سه پردازه در سیستم عامل موجود هستند و تکه‌ی تخصیص داده شده به هر پردازه در جدول زیر نمایش داده شده است.

Process	Limit	Relocation
A	16	40
B	30	0
C	20	70

- دقت کنید که هر پردازه مقدار Limit و Relocation خودش را دارد؛ در هنگام تعویض متن، مقدار این دو رجیستر با مقدار مناسب پردازه پر می‌شود.
- جدول زیر به عنوان نمونه چند آدرس منطقی را به آدرس فیزیکی نگاشت می‌کند.

Process	Logical Address	Physical Address
A	12	52
B	2	2
A	20	Trap
C	0	70

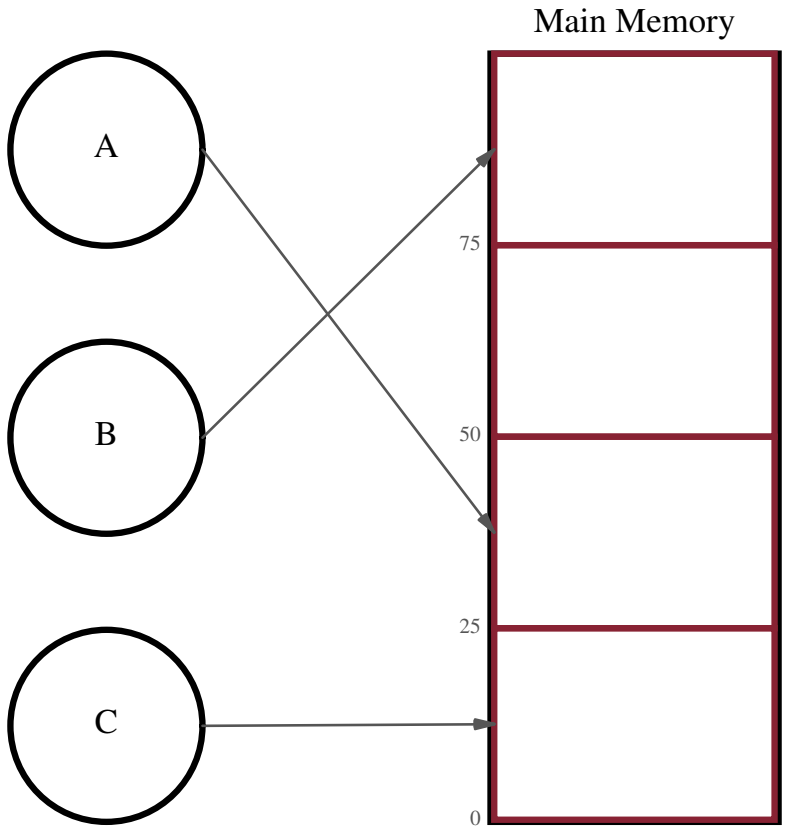
اندازه‌ی تکه‌های حافظه

در تخصیص حافظه‌ی پیوسته به پردازه‌ها، اندازه‌ی تکه‌های حافظه به دو شکل انجام می‌شود.

الف) تکه‌های برابر: حافظه به تکه‌های مساوی تقسیم می‌شود و به هر پردازه یک تکه تخصیص می‌یابد.

ب) تکه‌های نابرابر: تکه‌های تخصیص یافته به پردازه‌ها ممکن است نابرابر باشند.

- در تخصیص حافظه‌ی پیوسته به صورت برابر، حافظه به تعدادی تکه‌ی برابر تقسیم می‌شود و به هر پردازه یکی از قسمت‌ها تخصیص می‌یابد.
- شکل زیر را در نظر بگیرید.

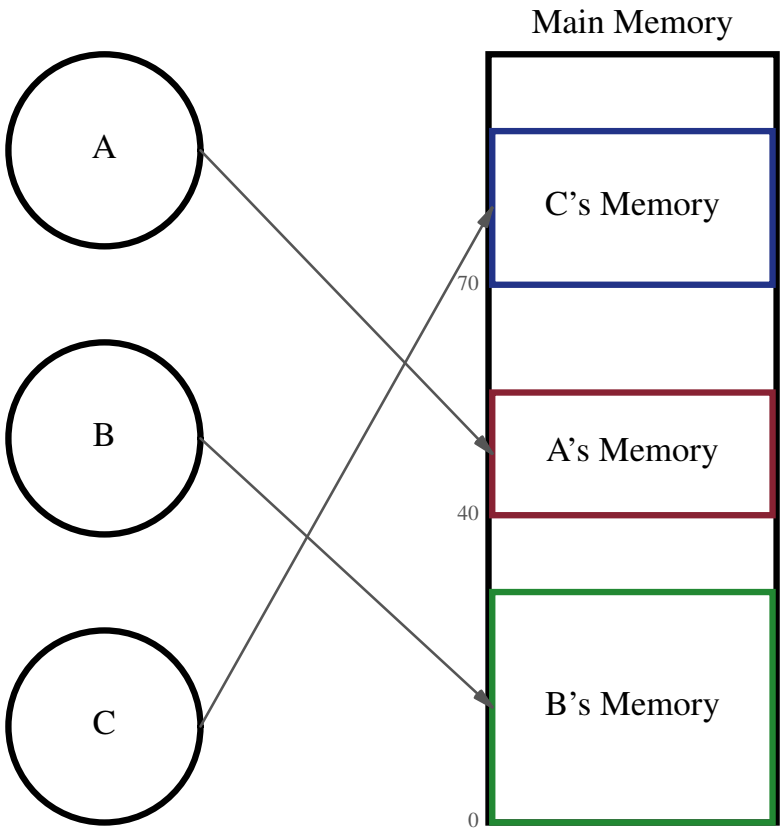


- جدول زیر مربوط به تخصیص حافظه به شکل بالا است (با فرض اینکه اندازه‌ی حافظه صد بایت باشد و حافظه به چهار قسمت تقسیم شده باشد).

Process	Limit	Relocation
A	25	25
B	25	75
C	25	0

- حتی اگر پردازه‌ای مقدار کمتری حافظه نسبت به اندازه‌ی تکه‌ها درخواست کند، سیستم عامل یک تکه‌ی کامل را در اختیار پردازه قرار می‌دهد.
- بنابراین، حافظه در داخل تکه‌هایی که به پردازه‌ها داده می‌شود هدر می‌رود. به این اتفاق چند پارگی داخلی یا Internal Fragmentation گفته می‌شود.

- در تخصیص حافظه‌ی پیوسته به صورت نابرابر، به هر پردازه با توجه به نیازش تکه‌ای از حافظه تخصیص می‌یابد.
- شکل زیر یک نمونه را نشان می‌دهد.

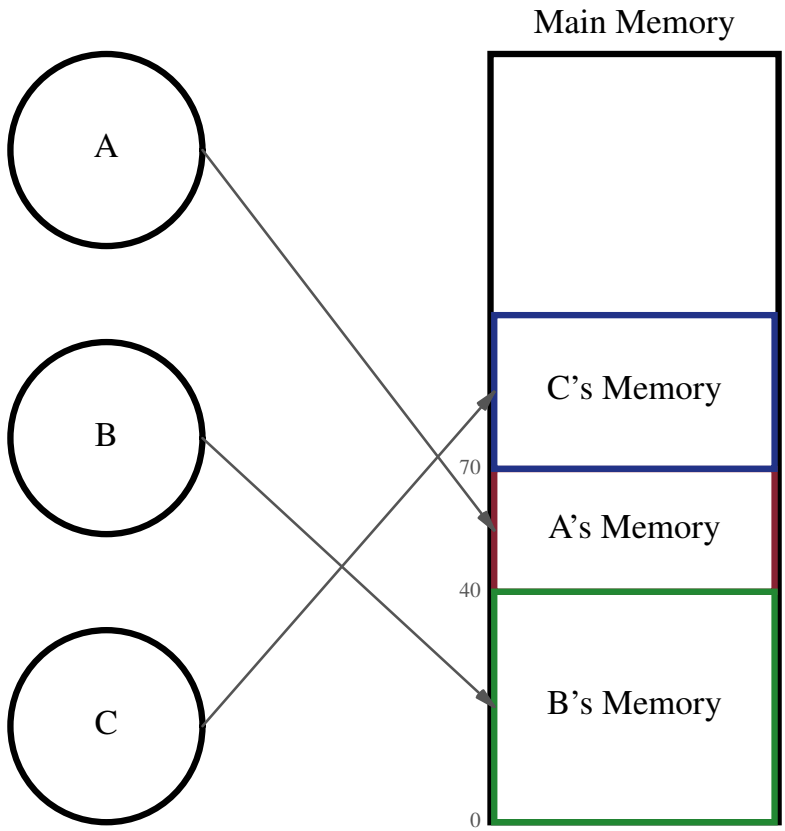


- جدول زیر مربوط به شکل بالا است (اندازه‌ی حافظه صد بایت است).

Process	Limit	Relocation
A	16	40
B	30	0
C	20	70

- شکل صفحه‌ی قبل را در نظر بگیرید.
- فرض کنید بخواهیم به پردازه‌ای بیست بایت حافظه تخصیص دهیم.
- اگر چه ۳۴ بایت از حافظه خالی است نمی‌توانیم این کار را انجام دهیم، چون حافظه‌های خالی پیوسته نیستند.
- به این اتفاق چند پارگی خارجی یا External Fragmentation گفته می‌شود.

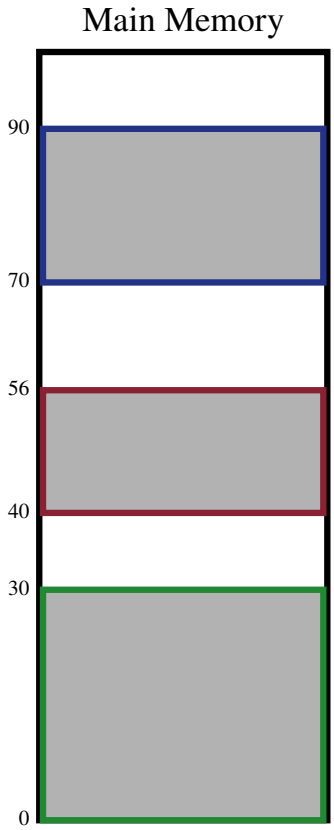
- یک راه برای حل چند پارگی خارجی، انتقال دادن همه‌ی تکه‌های تخصیص یافته‌ی حافظه به یک سمت از حافظه است تا تکه‌های حافظه‌ی تخصیص نیافته کنار هم قرار گیرند.
- به این کار Compaction گفته می‌شود.
- اگر Compaction برای مثال قبل اجرا شود، تخصیص حافظه به صورت زیر تغییر می‌کند.



- جدول نیز به صورت زیر تغییر می‌کند.

Process	Limit	Relocation
A	16	30
B	30	0
C	20	46

- وضعیتی از حافظه را در نظر بگیرید.
- برای تخصیص حافظه باید تکه‌ی تخصیص نیافته‌ای از حافظه انتخاب شود که به اندازه‌ی کافی بزرگ باشد.
- برای مثال در شکل زیر، برای تخصیص یک تکه از حافظه با اندازه‌ی ده، می‌توان از تکه‌های خالی که از آدرس سی، آدرس پنجاه و شش یا آدرس نود شروع می‌شود انتخاب کرد.



- سه راه ساده برای انتخاب تکه‌ی خالی موارد زیر هستند:

First-fit: اولین تکه‌ی خالی از حافظه که به اندازه‌ی کافی بزرگ باشد انتخاب می‌شود.

Best-fit: کوچک‌ترین تکه‌ی خالی از حافظه که به اندازه‌ی کافی بزرگ باشد انتخاب می‌شود.

Worst-fit: بزرگ‌ترین تکه‌ی خالی از حافظه که به اندازه‌ی کافی بزرگ باشد انتخاب می‌شود.

در تخصیص حافظه‌ی ناپیوسته امکان دارد بیش از یک تکه از حافظه به هر پردازه تخصیص داد.

- مشابه حالت پیوسته، تخصیص حافظه در حالت ناپیوسته به دو شکل انجام می‌شود:

  الف) تکه‌های نابرابر: به این حالت قطعه‌بندی (Segmentation) گفته می‌شود.

  ب) تکه‌های برابر: به این حالت صفحه‌بندی (Paging) گفته می‌شود.

- این دو را بررسی خواهیم کرد.

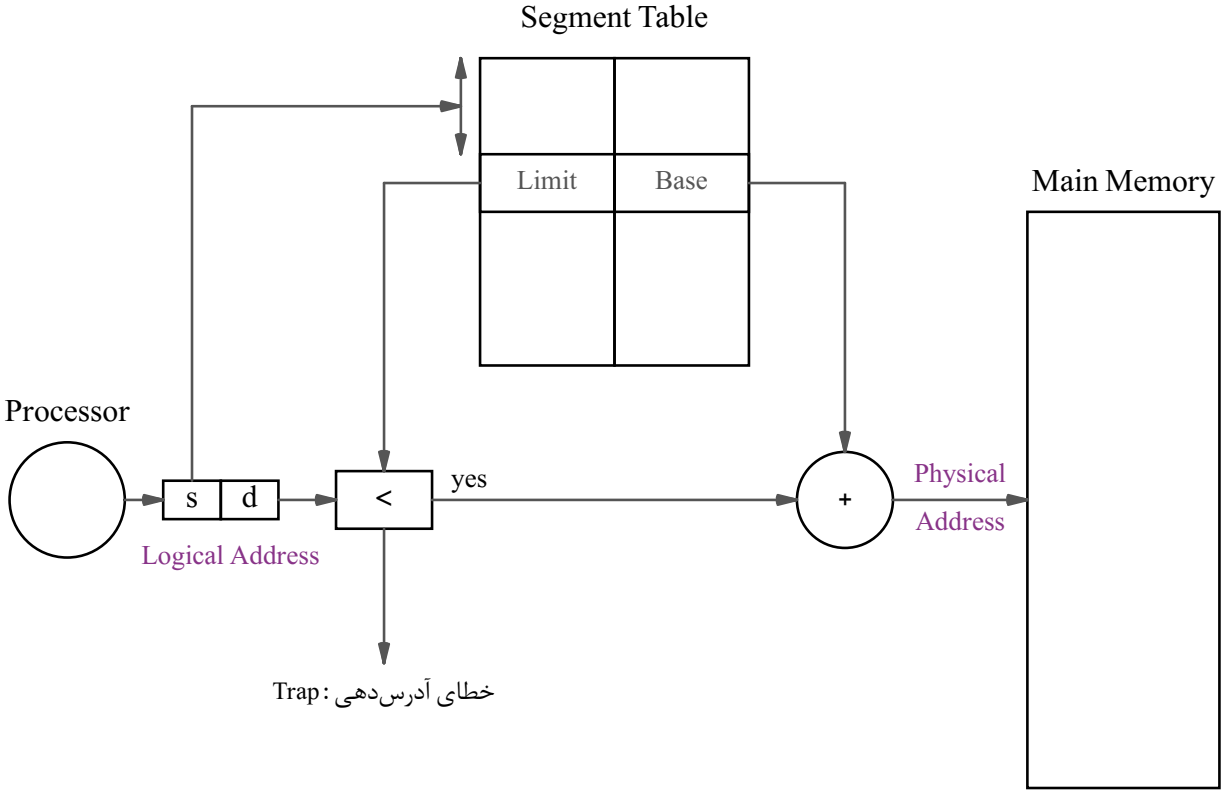
- حافظه‌ی یک پردازه به قسمت‌هایی مثل Stack، Heap، Data یا کد مربوط به کتابخانه‌های مختلف تقسیم می‌شود.
- در قطعه‌بندی (Segmentation) به هر یک از این قسمت‌ها یک قطعه (Segment) از حافظه تخصیص می‌دهیم.
- برای هر پردازه اطلاعات این قطعه‌ها را در جدولی به نام جدول قطعه (Segment Table) قرار می‌دهیم.
- مثال:

Segment ID	Segment Limit	Segment Base
0	16	40
1	30	0
2	20	70

- دقت کنید که هر پردازه یک جدول قطعه‌ی مختص خودش دارد.
- آدرس منطقی در قطعه‌بندی از دو قسمت تشکیل شده است. آدرس منطقی به صورت **(s, d)** هست که در آن **s** شناسه‌ی قطعه و **d** شماره‌ی بایت قطعه (offset) را مشخص می‌کند.
- برای نمونه در جدول بالا آدرس منطقی **(2, 8)** به بایت هشتم قطعه‌ی دوم اشاره می‌کند. بنابراین آدرس فیزیکی معادل آن برابر ۷۸ است.
- چند مثال دیگر را در جدول زیر می‌بینید.

Logical Address	Physical Address
(2, 8)	78
(1, 0)	0
(2, 24)	Trap
(0, 12)	52

شکل زیر روش کار قطعه‌بندی را به صورت گرافیکی نشان می‌دهد.



- با توجه به شناسه‌ی قطعه در آدرس منطقی، سطر مناسب جدول قطعه انتخاب می‌شود.
- با توجه به مقدار Limit در این سطر، بررسی می‌شود که آدرس از قطعه بیرون نباشد.
- سپس با جمع کردن قسمت دوم آدرس منطقی و مقدار Base آدرس فیزیکی محاسبه می‌شود.