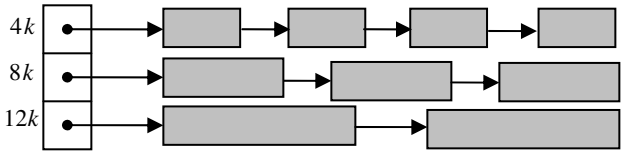
جلسه دهم:

4- **بدترین مناسب**(*worst – fit*):

در این الگوریتم بزرگترین حفره انتخاب شده و پردازه در آن قرار میگرد. دلیل انتخاب بزرگترین حفره این است که از فضای باقی مانده دیگر پردازه ها می توانند استفاده کنند.ایراد این تکنیک این است که امکان دارد، تقاضاهای که ناحیه بزرگی می خواهند، دیگر نتوانند برآورده شوند چرا که بلاک های بزرگ زودتر تخصیص یافته و کوچک می شوند.

5- **سریعترین مناسب**(*Quick Fit*):

در این الگوریتم لیستی از اندازه پردازه های متداول تهیه می شود و آرایه ای با n خانه در نظر گرفته می شود که هر خانه این آرایه شامل یک اشاره گر به ابتدای لیست یک فضای خالی به اندازه متداول است به عنوان مثال فضا های متداول می توانند $2k$, $4k$, $8k$, $12k$ *و ... باشند که برای هر کدام یک خانه آرایه در نظر گرفته می شود.عیب این روش این است که اگر پروسسی خاتمه یابد باید فضای آزاد شده آن به لیست مناسب اضافه شود که این کار زمانبر می باشد.*



6- **الگوریتم رفاقتی**(*Buddy*):

در این روش حفره ها(فضا های خالی) به صورت توان های 2 در نظر گرفته می شود.به عنوان مثال حفره هائی به اندازه $1k$, $2k$, $4k$, $8k$, $16k$, $32K$... *و برای هر گروه یک لیست جداگانه در نظر گرفته می شود.بدین ترتیب جهت تخصیص یک بلاک تنها باید بلاک مورد نظر را از لیست مناسب خارج کرد. پس از تخصیص اگر فضای باقی مانده آن بلاک، توانی از 2 باشد در لیست مربوطه اش قرار می گیرد و در غیر این صورت به چندین بخش که اندازه هر کدام توانی از 2 می باشد تقسیم میشود. از طرف دیگر در این روش بلوک های کنار هم می توانند باهم ترکیب شده و بخش بزرگتری را پدید بیاورند.*

مثال. با قسمت هائی از حافظه به اندازه هائی $100k$, $500k$, $200k$, $300k$, $600k$ *هر یک از روش های اولین جای مناسب، بهترین جای مناسب و بدترین جای مناسب پردازه هائی با اندازه* $212k$, $417k$, $112k$, $426k$ *را چگونه در حافظه قرار می دهند و کدام روش از حافظه به طور بهینه استفاده می کند. ترتیب ورود پردازه ها را یک بار از راست به چپ و یک بار از چپ به راست بگیرید.*

$H: 100k , 500k , 200k , 300 , 600k$
$P: 212k , 417 , 112k , 426k$

الف. اولین مناسب: *در حالت از چپ به راست پردازه 426 باید منتظر بماند*

$100k , 176k , 200k , 300k , 183k ,$ *لیست فضای آزاد، ترتیب پردازه ها از چپ به راست*
$100k , 74k , 88k , 88k , 183k$ *لیست فضای آزاد، ترتیب پردازه ها از راست به چپ*

ب. بهترین مناسب

$100\ k , 83\ k , 88\ k , 88\ k , 174\ k ,$ *لیست فضای آزاد، ترتیب پردازه ها از چپ به راست*
$100\ k , 74\ k , 88\ k , 88\ k , 183\ k$ *لیست فضای آزاد، ترتیب پردازه ها از راست به چپ*

ب.بدترین مناسب

$100\ k , 388\ k , 200\ k , 88\ k , 174\ k ,$ *لیست فضای آزاد، ترتیب پردازه ها از چپ به راست*
$100\ k , 83\ k , 200\ k , 188\ k , 388\ k$ *لیست فضای آزاد، ترتیب پردازه ها از راست به چپ*

روش بهترین مناسب از حافظه به صورت بهینه استفاده می کند. زیرا برای تمام پردازه ها فضای لازم را پیدا می کند و پارگی خارجی در آن حداقل است.

مثال. 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 $20k$, $30k$, $20k$, $35k$ *به ترتیب از راست به چپ ذکر شده داده شود و از روش Next Fit استفاده شود و تخصیص از اول حافظه شروع شود، وضعیت حافظه را بعد از این تخصیص ها مشخص کنید.*

$40k , 25k , 45k , 50k , 60k , 40k \rightarrow$ *شروع*

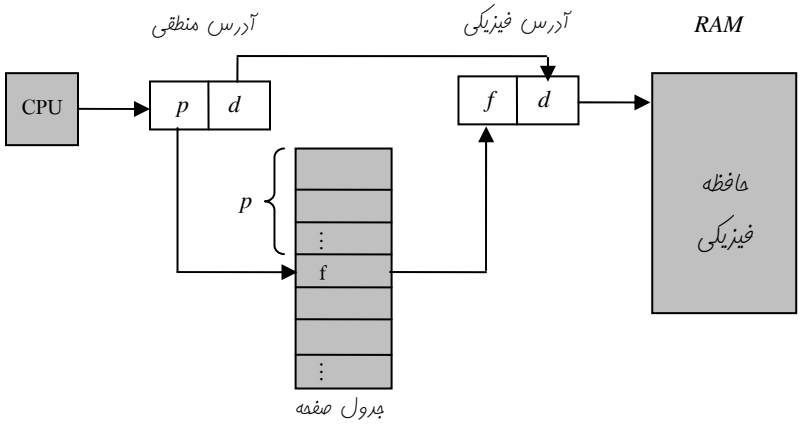
جواب: $20k , 25k , 15k , 30k , 25k , 40k$

صفحه بندی(*Paging*):

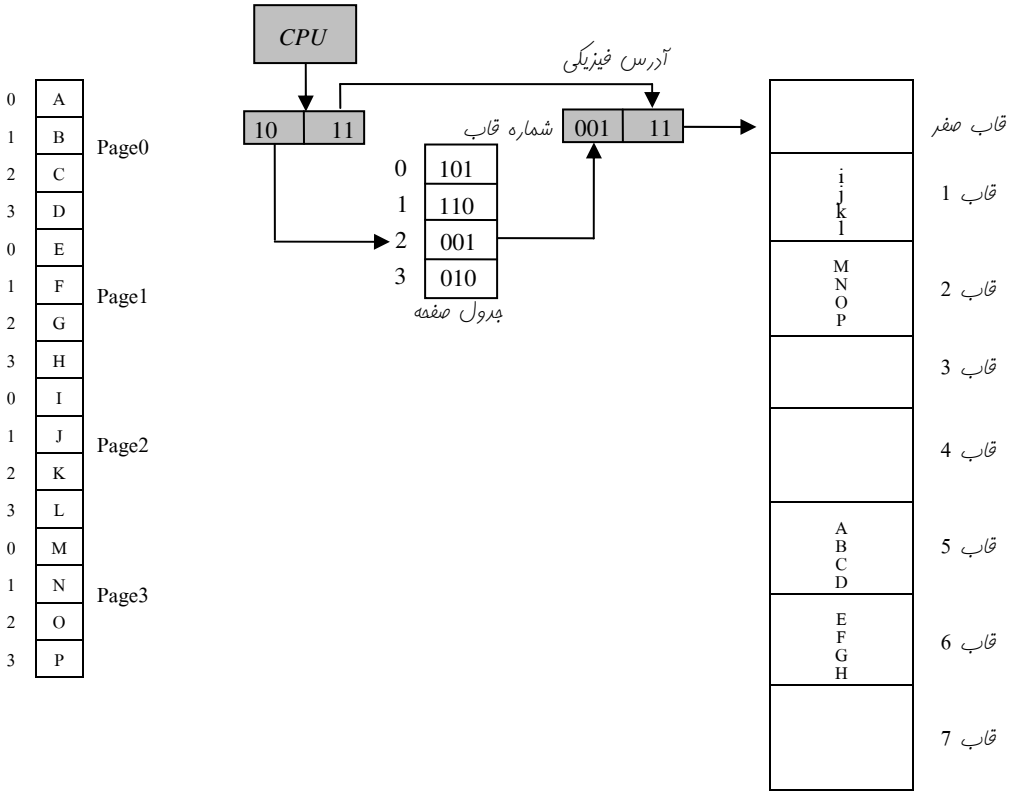
*در روش صفحه بندی برنامه کاربر(فضای آدرس منطقی) به بخش هائی به اندازه ثابت به نام صفحه(**Page**) تقسیم می شود و حافظه فیزیکی به قسمت هائی به نام قاب(**Frame**) تقسیم می شود که اندازه هر فریم برابر اندازه **Page** می باشد. صفحه بندی این امکان را می دهد که قسمت های یک برنامه (**Page** های هر برنامه) در حافظه پراکنده باشند(لازم نیست مجاور هم باشند)*

نحوه تبدیل آدرس منطقی به آدرس فیزیکی:

*در این تکنیک هر آدرس تولید شده توسط **CPU**(یعنی آدرس منطقی) از دو بخش شماره صفحه(**p**) و افست صفحه(**d**) تشکیل شده است. شماره صفحه به عنوان اندیس جدول صفحه(**page table**) استفاده می گردد. جدول صفحه شامل آدرس مبنای هر صفحه در آدرس فیزیکی **RAM** است. این آدرس مبنا با آدرس افست منطقی ترکیب شده و آدرس فیزیکی نهائی را تشکیل می دهد. شکل زیر این موضوع را نشان می دهد.*



مثال. فرض کنید تعداد صفحات $4 = 2^2$ عدد و تعداد قاب ها $8 = 2^3$ عدد باشد.



□ به هر کدام از سطر های جدول صفحه یک مدخل یا Entry گویند. که هر مدخل شامل شماره یک فریم و اطلاعات دیگری از قبیل بیت معتبر(نامعتبر)، بیت Read، بیت write و بیت های دیگر می باشد.

$\lceil Log_2^{\text{تعداد صفحات}} \rceil$ = تعداد بیت های مورد نیاز برای شماره جدول صفحه(p)

حافظه منطقی ≥ حافظه فیزیکی $\Rightarrow$ تعداد ***page*** ها ≥ تعداد فریم

$\lceil Log_2^{\text{اندازه صفحات}} \rceil$ = تعداد بیت های مورد نیاز برای افست(d)

□در صفحه بندی اگر بیت ***valid*** صفر باشد بدین معناست که آدرس صفحه ذکر شده نامعتبر می باشد یا چنین صفحه ای وجود ندارد.