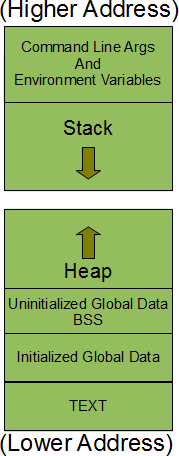
برای فهم بهتر نخ ابتدا باید فرایند و تعویض زمینه را متوجه شویم. اما با معرفی سیستم عامل و کرنل آن را شروع می‌کنیم.

سیستم عامل چیست؟ سیستم عامل یک واسط بین کاربر کامپیوتر و سخت افزار است. این روز ها برای اجرا بیشتر برنامه ها فقط کافیت یا روی آیکون آن‌ها کلیک کنیم و یا دستور آن‌ها را در ترمینال وارد کنید تا برنامه بصورت جادویی برای ما اجرا شود ولی همین کار ساده که به راحتی کلیک کردن یا تایپ کردن چند کرکتردر ترمینال است پروسه چندان بلندی را طی می‌کند. بصورت وقتی را تصورکنید که کامیپوتر را روشن و قصد اجرای برنامه Microsoft Word را دارید، بصورت معمول این برنامه بر روی حافظه ثانویه شما (که میتواند دیسک سخت یا درایو حالت جامد باشد) قرار دارد. این نسخه از برنامه حالت Passive نرم‌افزار است که بعد از اجرای آن فرایندی ساخته می‌شود که این نسخه Active نرم‌افزار است. برای ساخت شدن یک فرایند از روی یک برنامه ای که روی حافظه ثانویه است دارای مراحلی است. ابتدا باید یک ساختار داده فرآیند در حافظه اصلی ساخته شود در ادامه کد های برنامه از روی حافظه ثانویه به حافظه اولیه انتقال پیدا کنند و در قسمت کد فرآیند قرار بگیرند، کتابخانه‌های مورد نیاز برنامه سپس در حافظه لود می‌شوند و حالت برنامه در حالت آماده اجرا قرار میگیرد. به محض اینکه کرنل اجازه اجرا شدن بدهد ابتدا وضعیت آن به وضعیت در حال اجرا تغییر خواهد کرد و سپس کد برنامه شروع به اجرا شدن می‌کند. تمامی مراحل ذکر شده توسط سیستم عامل فقط با یک کلیک انجام می‌شود تا دردسر کاربر نهایی به حداقل برسد.



وظایف اصلی سیستم عامل:

* مدیریت حافظه
* مدیریت فرآیندها
* مدیریت دستگاه‌ها
* مدیریت فایل‌ها
* امنیت
* کنترل عملکرد سیستم
* حسابداری وظایف
* شناسایی خطا ها

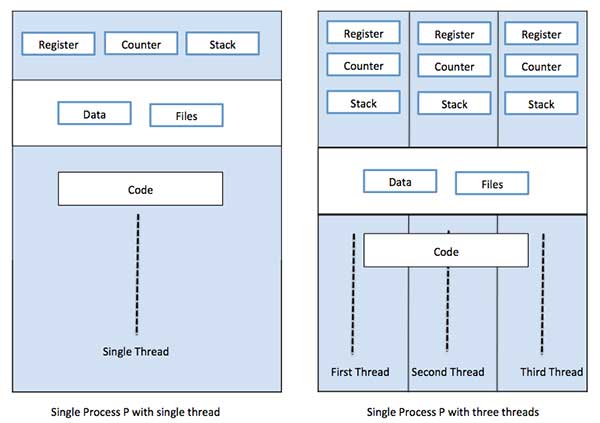
# فرآیند چیست؟

فرآیند یک نمونه لود شده از برنامه کامیپوتر در حافظه است. فرایند یک ساختار داده است که شامل قسمت های متفاوتی از جمله حافظه مورد استفاده خودش، پشته، هرم، کد نوشته شده برنامه و یک حالت است که میتواند مقادیر متعددی مثل در حال اجرا در حال خواب یا در بلاک شده و ... باشد و تغییر حالت فرایند ها بر عهده کرنل است.

# ترد چیست؟

ترد چیزی نیست به جز یک "زمینه اجرا"، حالا این زمینه اجرا چیه؟ اگر مصرف کوک رو دیده باشید میبینید که ابتدا کوک را روی سطح صافی میریزند و از آن یک خط صاف درست میکنند. الان تا اینجای کار ما یک ترد ساخته ایم در واقع ما یک زمینه اجرا ساخته ایم که فرد استعمال کننده میتواند به راحتی فقط با یک نقس تمام کوک را اسنیف کند. در دنیای کامپیوتر استعمال کننده کسی نیست جز CPU.

به عنوان مثالی دیگر فرض کنید در حال خواندن یک کتاب هستید، کتاب را تا جایی پیش برده اید ولی میخواهید به کار دیگر برسید میتوانید با نوشتن سه عدد رو یک برگه کاغذ سری بعدی متوجه بشید که از کجا شروع کنید، با یادداشت شماره صفحه، شماره خط و شماره کلمه در خط. این سه عدد در واقع همان زمینه اجرا هستند چون در دفعات بعد که میخواستیم کتاب را بخوانیم میتوانیم سریع بفهیم پروسه کتاب خواندن را تا کجا اجرا کرده بودیم و میتوانیم کار را ادامه دهیم. اگر یک دوست داشته باشید و اوبخواهد در زمانی که شما کتاب را نمیخوانید خودش هم کتاب را بخواند میتواند با ذخیره این سه عدد سری بعدی ادامه کتاب را بخواند بدون اینکه نشانه گذاری شما آسیبی ببیند یا تصادمی پیش بیاید. پس اگر دقت کرده باشید در این جا ما یک کتاب داریم ولی میتوانیم چندین زمینه اجرایی داشته باشیم در کامیپوتر هم همینگونه است هر برنامه ای که نوشته میشود یک قطعه کد دارد ولی میتواند چندین زمینه اجرا داشته باشد. عکس زیر مبین همین واقعیت است.



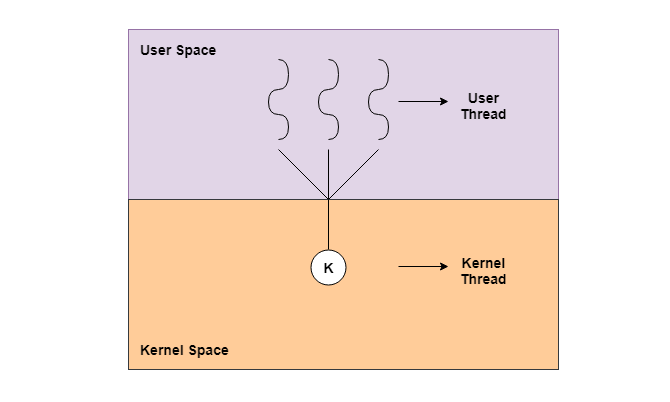
اگر فرض کنیم هر هسته میتواند در لحظه یک کار را انجام دهد پس در کامیپوتر های قدیمی تک هسته ای این اجرا شدن موازی برنامه ها، مثل پخش موسیقی در پس زمینه و کد زدن همراه آن، چگونه انجام میشوند؟ در واقع CPU به ما توهم موازی سازی میدهد، در کامیپوتر های تک هسته ای عملا چیزی به نام اجرای موازی نداریم بلکه مفهوم به نام همروندی وجود دارد. CPU در هر لحظه بین کار های مختلف سوییچ میکند و یک تکه از هر کدام را اجرا میکند اما این تعویض آنقدر سریع اتفاق میفتد که کاربر نهایی عملا متوجه این وقفه در اجرا نمیشود، دقیقا به همان صورت که شما میتوانید یک کتاب را با دوستتان به اشتراک بگذارید تسک ها میتوانند CPU را با هم بصورت اشتراکی مصرف کنند. هر بار موقع تعویض برنامه و اجرای برنامه دیگر تمام چیز های مورد نیاز برای از سر گیری کد را در زمینه اجرا آن ذخیره میکند و سپس شروع به اجرای برنامه دیگری میکند.

اگر بخواهیم تعریف فرآیند را از دید دیگری بیان کنیم باید بگوییم فرآیند چیزی نیست به جز یک تعدادی از منابع که به یک محاسبات(Computation) تخصیص داده شده است (منابع مثل: حافظه، اطلاعات فایل های باز شده یا Connection های برقرار شده (File descriptors)، اعتبارات امنیتی (مثل اینکه کدام یوزر فرآیند را اجرا کرده است).

پس فرآیند کاملا متفاوت از نخ است. هر فرآیند میتواند یک یا چندین نخ داشته باشد و عملا چیزی به جز تخصیص منابع به محاسبات نیست ولی نخ در واقع یک زمینه اجرایی است.

# انواع نخ ها:

دو نوع نخ داریم یکی نخ های سطح کاربر و نخ های سطح سیستم عامل یا همان کرنل.



# نخ های سطح کاربر:

نخ های سطح کاربر توسط برنامه نویس مدیریت میشوند و کرنل از وجود آنها مطلع نیست، دیدی که سیستم عامل از آنها دارد انگار که با یک فرآیند تک نخه سر و کار دارد. نخ های سطح کاربر نسبت به نخ های سطح هسته سبک تر و سریعتر هستند همچنین برای همگام کردن نخ های سطح کاربر سیستم عامل هیچ کاری نمیتواند بکند.

مزایا:

* نخ های سطح کاربر راحت تر ساخته و مدیریت میشوند.
* نخ های سطح کاربر در هر سیستم عامل میتوانند اجرا شوند.
* برای تعویض زمینه ها یا همان ترد ها نیاز به اجازه از سیستم عامل نیست.

بدی‌ها:

* برنامه های چند نخی در سطح کاربر نمیتوانند از موازی سازی واقعی استفاده کنند.
* اگر یک نخ یک فرآیند بلاک کننده اجرا کند کل فرآیند بلاک میشود.

# نخ های سطح هسته:

نخ های سطح هسته بصورت مستقیم توسط سیستم عامل هندل میشوند. اطلاعات مربوط به نخ و اطلاعات زمینه همه توسط خود هسته مدیریت میشود بخاطر همین موضوع نخ های سطح کاربر آهسته تر هستند.

مزایا:

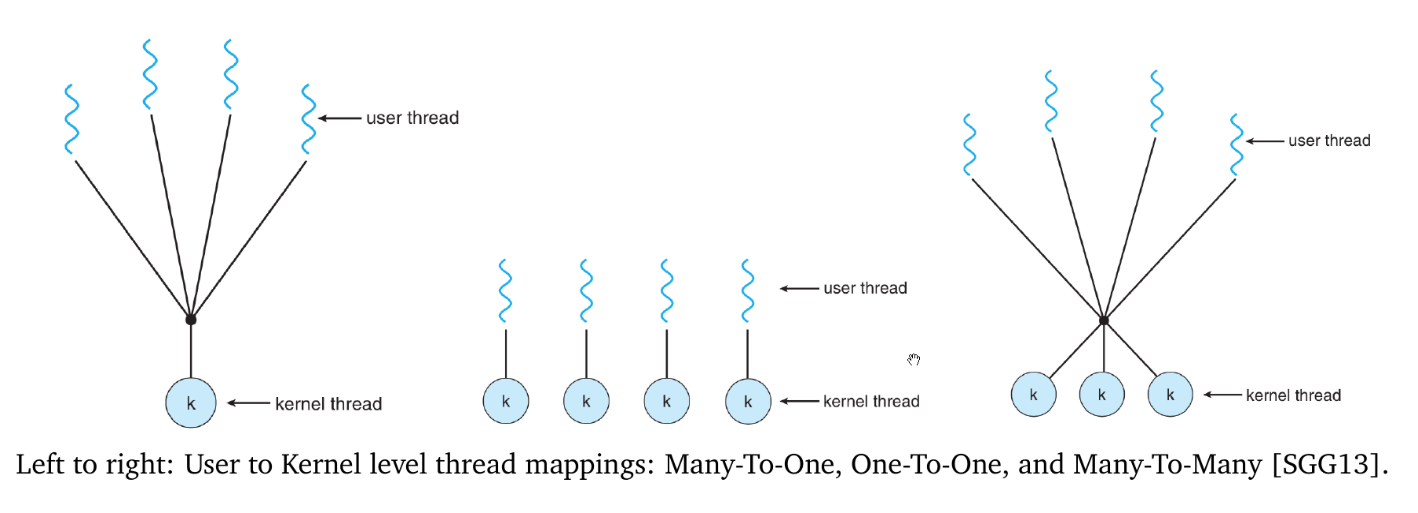
* هر کدام از ترد های برنامه ای که از ترد های سطح هسته استفاده میکنند میتوانند روی یک هسته جدا اجرا شوند. (موازی سازی واقعی)
* خود روتین های سیستم عامل میتوانند چند نخه شوند.
* اگر یک نخ از یک پروسه بلاک شد، نخ دیگر میتواند بدون بلاک شدن به کار خود ادامه دهد.

بدی ها:

* برای سوییچ شدن دو ترد از ترد های یک برنامه نیاز به دخالت سیستم عامل است.
* کرنل های سطح هسته کند تر ساخته میشوند و در مقایسه با سطح کاربر مدیریت کند تری دارند.

انواع نگاشت های کرنل های سطح کاربر به سطح هسته:

توضیح شکل زیر...



مقایسه نهایی نخ های سطح کاربر و هسته:

