نام :عرفان ملاباقری

مقدمه:

تعریف یک متغیر، ساخت instance، توابع، ایجاد یک thread و ... در برنامه نیاز به ذخیره سازی در حافظه دارند و همه آن ها فضایی از حافظه را اشغال می کنند. بنابراین لازم است تا یک برنامه نویس درک درستی از حافظه ای که در اختیار می گیرد داشته باشد.

حافظه heap:

حافظه heap یک بخش از حافظه کامپیوتر است که برای ذخیره داده‌ها و آبجکت‌هایی که به صورت پویا تخصیص می‌یابند، استفاده می‌شود. در برخی از زبان‌های برنامه‌نویسی، مانند C++ و Java، برای ایجاد و مدیریت آبجکت‌ها و داده‌ها در زمان اجرا از حافظه heap استفاده می‌شود.

حافظه Heap در قست user-space حافظه مجازی قرار دارد و به صورت دستی توسط برنامه نویس مدیریت می شود. Heap مربوط به زمان اجرا (runtime) است و فضای اشغال شده در heap با اتمام کار تابع آزاد نمی شوند و تا زمانی که Garbage Collector این فضا را آزاد کند یا توسط برنامه نویس داده ها از حافظه heap پاک نشوند در این فضا باقی می ماند. اندازه حافظه heap متغیر است به همین دلیل به آن dynamic memory گفته می شود.

حافظه stack:

حافظه stack یک بخش از حافظه کامپیوتر است که برای ذخیره داده‌ها و اطلاعات مربوط به اجرای فرآیندهای برنامه استفاده می‌شود. در حافظه stack، اطلاعات به صورت پشته‌ای (LIFO - Last In, First Out) ذخیره می‌شوند، به این معنی که آخرین داده‌هایی که وارد شده‌اند، اولین داده‌هایی هستند که خارج می‌شوند. این حافظه برای ذخیره متغیرهای محلی، آدرس بازگشت از توابع، و اطلاعات مربوط به فرآیندهای فعلی استفاده می‌شود.

در بخش user-space حافظه قرار دارد و به صورت خودکار توسط CPU مدیریت می شود. متغیرهای غیر استاتیک، پارامتر های ارسالی به توابع و آدرس های مربوط به return توابع در این حافظه ذخیره می شوند. اندازه حافظه stack ثابت است به همین دلیل به آن static memory گفته می شود.

تفاوت بین حافظه heap و حافظه stack :

تفاوت اصلی بین حافظه heap و حافظه stack در روش ذخیره و مدیریت داده‌هاست:

1 .مدیریت:

حافظه heap : داده‌ها در حافظه heap به صورت داینامیک تخصیص می‌یابند و باید به صورت دستی مدیریت شوند. این به این معنی است که باید داده‌ها را ایجاد کرد، حذف کرد و یا به صورت پویا تغییر داد.

حافظه stack : داده‌ها در حافظه stack به صورت خودکار مدیریت می‌شوند. هنگامی که یک تابع فراخوانی می‌شود، فضای لازم برای متغیرهای محلی آن تابع اختصاص می‌یابد و هنگامی که تابع به پایان می‌رسد، فضای مربوط به آن تابع از حافظه آزاد می‌شود.

2 .زمان زنده ماندن:

حافظه heap : داده‌های موجود در حافظه heap تا زمانی که به آنها اشاره‌ای وجود داشته باشد، زنده می‌مانند. به عبارت دیگر، زمان زنده ماندن آنها تا زمانی است که متغیرها یا آبجکت‌هایی که به آنها اشاره دارند، وجود داشته باشند.

حافظه stack : داده‌های موجود در حافظه stack فقط تا زمان اجرای مربوطه فرآیندهای برنامه‌ای (مانند تابع‌ها) زنده می‌مانند. بنابراین، هنگامی که یک تابع از اجرای خود خارج می‌شود، داده‌های مربوطه از حافظه حذف می‌شوند.

3 .محدودیت‌ها:

حافظه heap : از نظر محدودیت‌ها، حافظه heap معمولاً بزرگتر و پر کنترل‌تر از حافظه stack است. این به این معنی است که ممکن است به میزان بیشتری از حافظه سیستم دسترسی داشته باشید، اما مدیریت دستی برای این محدوده ضروری است.

حافظه stack: اندازه حافظه stack معمولاً محدودتر است و به طور خودکار توسط سیستم عامل مدیریت می‌شود. این به این معنی است که ممکن است در صورت استفاده از زیادی فضا یا رشد زیاد در عمق توابع، با خطاهای مانند Stack Overflow مواجه شوید.

منابع:virgool.io و chatgpt