دو: آرایه

ساختمان داده ها و الگوریتم

مدرس: دکتر نجمه منصوری

نگارنده: سجاد هاشمیان

۱. آرایه چیست؟

آرایه تعدادی متغیر از یک نوع داده و تحت یک نام می‌باشد. هر یک از متغیرهای درون آرایه با یک شماره که به آن «اندیس» می‌گوییم از یکدیگر متمایز می‌شوند. متغیرهای درون آرایه را «عناصر آرایه» می‌نامند که همگی قابلیت نگهداری فقط یک نوع داده را دارند. عناصر درون آرایه از نظر فیزیکی مکان‌های متوالی در حافظه اصلی رایانه را اشغال می‌کنند. بنابراین تعداد عناصر درون آرایه محدود و ثابت می‌باشد. اما از نظر منطقی عناصر درون آرایه را می‌توانند به صورت یک سطر یا یک ستون (در آرایه یک بعدی) یا به صورت یک جدول یا ماتریس (در آرایه دو بعدی) یا در داخل یک مکعب در آرایه سه بعدی تصور شوند؛ یا حتی در ابعاد بیشتر که از این نظر محدودیتی وجود ندارد. بردار یک آرایه یک بعدی است و ماتریس یک آرایه دوبعدی است که شامل چند سطر و ستون است. آرایه سه بعدی شامل سطری از سطح‌ها و ستون‌ها است. به همین ترتیب آرایه‌ای با ابعاد بیشتر را می‌توان با آرایه‌ای باابعاد کمتر ایجاد کرد؛ به شهود ساده تر آرایه در واقع **تابع تناظری از اندیس های عددی به آدرس‌های حافظه** است.

۱.۱ آدرس دهی در آرایه ها

به طور کلی آدرس‌دهی خانه ها در آرایه‌ها با توجه به وضعیت سطری یا ستونی آن آرایه تعیین می‌شود. یعنی اگر $A[(L_1, \dots, U_1), (L_2, \dots, U_2), \cdots, (L_n, \dots, U_n)]$ آرایه ای n-بعدی باشد، آنگاه آدرس خانه $A[x_1, x_2, \cdots, x_n]$ به شکل زیر بدست خواهد آمد:

وضعیت سطری :
$$A[x_1, x_2, \dots, x_n] = S + \left[\sum_{i=1}^{n} (x_i - L_i)\left(\prod_{j=i+1}^{n} U_i - L_i + 1\right)\right] \times B$$

وضعیت ستونی:
$$A[x_1, x_2, \dots, x_n] = S + \left[\sum_{i=n}^{1} (x_i - L_i)\left(\prod_{j=1}^{i-1} U_i - L_i + 1\right)\right] \times B$$

که در آن B به معنی مقدار حجم مورد نیاز برای ذخیره هر خانه آرایه و S به مثابه آدرس شروع اولین خانه آرایه است. میبیند که این آدرس دهی قابل تعمیم میباشد و ما در ساختن آرایه ها محدودیتی برای تعداد بعد ها نداریم، اما با افزایش بعد ها به علت نحوه آدرس‌دهی آرایه ها ممکن است که فاصله بین دو خانه که در اندیس گذاری با هم تفاوت چندانی ندارند فاصله بسیاری داشته باشند که این می‌تواند وابسته به سرعت سخت‌افزار پیمایش آرایه را کند و در نتیجه دسترسی به «عناصر آرایه» را مختل کند.

مثلا آدرس دو خانه $[x_1, x_2, \dots, x_n]$ و $[x_1, x_2 + 1,\dots,x_n]$ را محاسبه کنید.

۱.۲ درج و حذف عناصر در آرایه

فرض کنید که به ساختمان داده ای نیاز دارید که بتواند داده‌های جدیدی را به داده‌های ذخیره‌سازی شده اش اضافه کند، یا که به طور بلعکس، تعداد از رکورد های فعلی را حذف کند.

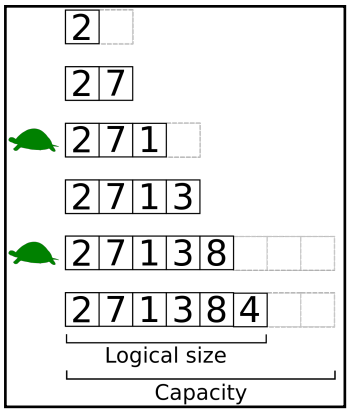
برای درج عنصری در آرایه تعدادی از عناصر باید به سمت پایین منتقل شوند تا عنصر جدید در محل مورد نظر قرار گیرد. اگر بخواهیم عنصر جدید در مکان kام آرایه درج شود کلیه عناصر از k به بعد باید شیفت داده شوند، سپس عنصر در مکان kام ذخیره شود.

به طور کلی $n - k + 1$ عنصر باید جابجا شوند. در بهترین حالت اگر عنصر جدید را در محل آخرین عنصر درج کنیم هیچ عنصری جابجا نمی‌شود. بدترین حالت زمانی اتفاق می‌افتد که بخواهیم عنصر جدید را در مکان اول آرایه درج کنیم در این حالت تعداد جابجایی‌ها دقیقا برابر با n می‌شود. به‌طور متوسط نیاز به $(n + 1)/2$ جابجایی است. با هربار عمل درج یک واحد به n تعداد عناصر آرایه اضافه می‌شود. n تعداد عناصری که در آرایه درج شده‌اند را نشان می‌دهد و ربطی به طول آرایه ندارد.

برای حذف هم باید عملیات مشابه‌ای را انجام داد و از این نتیجه می‌گیریم که برخلاف سرعت دسترسی به عناصر آرایه ($O(1)$)، عملیات درج و حذف در آرایه ها اصلا خوب نیست ($O(n)$) و از ثابت بودن تعداد خانه‌های آرایه ها که بگذریم، این ساختمان داده اصلا برای درج و حذف خوب نیست.

۲. انواع دیگر آرایه ها

۲.۱ آرایه های پویا



شکل بالا اضافه کردن چندین مقدار را به یک آرایه پویا نمایش می دهد. خانه های خاکستری برای گسترش رزرو شده اند.

ساده‌ترین آرایه پویا با اختصاص دادن آرایه‌ای به طول ثابت ساخته می‌شود که بعد آن را به دو قسمت تقسیم می‌کنند: قسمت اول عناصر آرایه پویا را ذخیره می‌کند و قسمت دوم ذخیره شده، یا بدون استفاده می‌ماند. سپس می‌توان با استفاده از فضای رزرو شده در زمانی ثابت عناصر را به انتهای آرایه پویا اضافه کرد یا آن‌ها را حذف کرد، تا زمانی که این فضا به‌طور کامل استفاده شود. تعداد عناصری که توسط محتویات آرایه پویا استفاده می‌شود اندازه منطقی یا اندازه آن است، در حالیکه اندازه آرایه زیرین ظرفیت آرایه پویا نام دارد، که حداکثر اندازه منطقی ممکن است. در برنامه‌هایی که اندازه منطقی کراندار است، این ساختار داده کفایت می‌کند. تغییر دادن اندازه آرایه زیرین عملی هزینه بر است، گاهی ناچار می‌شویم تمام محتویات آرایه را کپی کنیم.

۲.۲ آرایه های انجمنی

ویژگی مهم آرایه‌ها که تاکنون مطرح شد، روش دستیابی به یک عنصر خاص، ازطریق یک نوع شمارشی به نام اندیس است. عناصر آرایه برحسب این اندیس مرتب هستند و با استفاده از مقادیر اندیس می‌توان به هر عنصر آرایه دست یافت در بعضی از برنامه‌های کاربردی مطلوب است که از طریق نام (بدون استفاده از اندیس) بتوان به اطلاعات دست یافت.

البته عموما این آرایه ها را با استفاده از درهم سازی، درخت های Trie یا قرمز-سیاه (وابسته به نوع اندیس ها) پیاده سازی می‌کنند.

۳. کاربرد های آرایه ها

۳.۱ ماتریس های اسپارس

برای ذخیرهٔ یک ماتریس $m \times n$ می‌توان از یک آرایهٔ دو بعدی با m سطر و n ستون استفاده کرد. گروهی از ماتریس‌ها وجود دارند که به آن ماتریس خلوت یا اسپارس می‌گوییم. در این ماتریس‌ها اکثریت عناصر مقدار صفر دارند.از آنجایی‌که ماتریس‌های اسپارس در عمل زیاد استفاده می‌شوند و برخی موارد اندازه‌های آن‌ها بسیار بزرگ است می‌بایست روش بهینه‌تری را برای ذخیره آن‌ها در کامپیوتر ارائه کنیم . یک روش آن است که از یک آرایه دو بعدی با سه ستون استفاده کنیم . ستون‌های اول و دوم این آرایه موقعیت سطر و ستون مقادیر در ماتریس اسپارس را نشان می‌دهند و ستون سوم مقدار ذخیره شده در آن سطر و ستون را نشان می‌دهند. (تعداد سطرهای این آرایه به تعداد مقادیر ذخیره شده در ماتریس اصلی است.)

۳.۲ نمایش چندجمله‌ای‌ها به کمک آرایه

همه چندجمله‌ای‌ها را می‌توان به کمک آرایه‌ها پیاده‌سازی کرد. روش‌های مختلفی برای این کار وجود دارد. مثلاً می‌توان بزرگترین درجه‌ای که در چندجمله‌ای می‌تواند وجود داشته باشد به عنوان N در نظر گرفت، در این صورت می‌توان آرایه‌ای تعریف کرد که تعداد سلول‌های ان برابر با $N+1$ باشد. اگر درجه چندجمله‌ای را بدانیم می‌توان هر جمله را در آرایه پیاده‌سازی کرد. در واقع $A[i]$ ضریب x^{n-i} است. پس از ذخیره چندجمله‌ای‌ها در داخل آرایه‌ها می‌توان اعمالی مانند جمع چندجمله‌ای و ضرب چندجمله‌ای را انجام داد. برای مثال اگر چندجمله‌ای شما $x^{1000}+1$ باشد روش قبل برای ذخیره‌سازی چندجمله‌ای ممکن است مناسب نباشد. روش دیگری نیز برای ذخیره چندجمله‌ای‌ها وجود دارد؛ در این روش از یک آرایه با دو سطر و k ستون استفاده می‌شود که k تعداد کل جملات تشکیل دهنده چندجمله‌ای است. سطر اول نشان‌دهنده همه ضرایب است و سطر دوم نشان‌دهنده توان متناسب با هر ضریب است.

۳.۳ ذخیره سازی اعداد بزرگ

این نیز یک دیگر از کاربر های مهم آرایه هاست، میدانید با توجه به محدودیت‌ها نمی توان اعداد بزرگ تر از چندبیت را ذخیره کرد و عموما در عملیات هایی که با این اعداد سر و کار دارند با مشکل سرریز داده ها روبه رو می‌شویم، برای جلوگیری از این مشکلات کافیست تا یک آرایه به اندازه کافی بزرگ یک بعدی را در نظر بگیرید و در خانه $A[i]$ رقم iام عدد را ذخیره کنید، حال با همان روش های **دبستانی** عملیات های ریاضی را خودتان برای این اعداد پیاده‌سازی کنید.

این روش به Arbitrary-precision arithmetic یا BigNum مشهور است و در زبان برنامه نویسی Python به صورت عمومی استفاده می‌شود، در Java هم تحت java.math.BigInteger در دسترس است.

۴. برای مطالعه بیشتر

Bank, Randolph E., Andrew H. Sherman, and Alan Weiser. "Some refinement algorithms and data structures for regular local mesh refinement." Scientific Computing, Applications of Mathematics and Computing to the Physical Sciences 1 (1983) : 3-17.

۵. تمارین برنامه‌نویسی

- HackerEarth, Roy and Symmetric Logos
- کوئرا، کلاس کد!
- کوئرا، ضرب ماتریس‌ها