پنج: لیست پیوندی

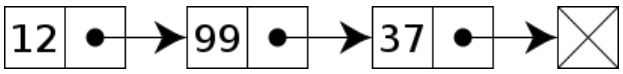
ساختمان داده ها و الگوریتم

مدرس: دکتر نجمه منصوری

نگارنده: سجاد هاشمیان

۱. لیست پیوندی چیست؟

لیست پیوندی ساختاری شامل دنباله‌ای از عناصر است که هر عنصر دارای اشاره‌گری به عنصر بعدی در دنباله است. لیست پیوندی از جملهٔ ساده‌ترین و رایج‌ترین داده‌ساختارها است و در پیاده‌سازی از داده‌ساختارها پشته (Stack) و صف (Queue) استفاده می‌شود. مزیت مهم لیست پیوندی نسبت به آرایه‌ها این است که ترتیب قرار گرفتن داده‌ها در آن با ترتیب قرار گرفتن آن‌ها در حافظه متفاوت است؛ به همین دلیل فهرست پیوندی دارای این ویژگی است که درج و حذف گره‌ها در هر نقطه‌ای از فهرست، با تعداد ثابتی از عملیات امکان‌پذیر است ولی از طرف دیگر لیست پیوندی اجازه دستیابی تصادفی به داده یا هرگونه اندیس‌گذاری را نمی‌دهد. در نتیجه بسیاری از اعمال ابتدایی نظیر به دست آوردن آخرین عنصر فهرست، پیدا کردن عنصر شامل داده مورد نظر، یا مشخص کردن مکان درج یک عنصر جدید ممکن است نیازمند بررسی اکثر عناصر فهرست باشد.



نام‌گذاری ها

هر رکورد از یک لیست پیوند داده شده اغلب " عنصر" یا " گره" نامیده می شود. فیلد هر گره که حاوی آدرس گره بعدی است معمولا "پیوند بعدی" یا " نشانگر بعدی" نامیده می شود. قسمتهای باقیمانده به عنوان قسمتهای "داده"یا "مقدار"شناخته می‌شوند. "سر" یک لیست اولین گره آن است. "دم" یک لیست ممکن است یا به بقیه لیست بعد از سر ، یا به آخرین گره در لیست اشاره داشته باشد.

معایب

- به علت وجود اشاره‌گرهای فراوان، لیست پیوندی از حافظه‌ی بیشتری استفاده می کند.
- عناصر لیست پیوندی پشت هم در حافظه ذخیره نمی‌شوند. این باعث می شود دسترسی به تک‌تک عناصر یا پردازش جمعی اطلاعات (مثل مقداردهی اولیه) کندتر باشد.
- لیست‌ها ارتباط تنگاتنگی با اشاره‌گرها دارند که این امر پیچیدگی برنامه و احتمال اشتباه را، به ویژه برای برنامه نویسان تازه‌کار بیش‌تر می‌کند.

۲. عملیات های لیست پیوندی

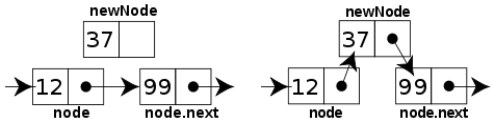
پیمایش لیست پیوندی

پیمایش یک لیست پیوندی به این صورت است که از گره اول شروع کرده و اشاره گر به عنصر بعدی را دنبال می‌کنیم تا به داده مورد نظر یا انتهای لیست برسیم.

```
node := list.firstNode
while node not null {
    (do something with node.data)
    node := node.next
}
```

درج گره در لیست پیوندی

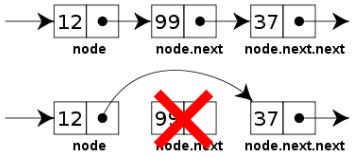
شبه کد زیر گره جدیدی را بعد از یک گره موجود داده شده درج می‌کند. شکل نحوهٔ درج را نشان می‌دهد.




```
function insertAfter(Node node, Node newNode) { // insert newNode after node
     newNode.next := node.next
     node.next    := newNode
 }
function insertBeginning(List list, Node newNode) { // insert node before current
first node
     newNode.next   := list.firstNode
     list.firstNode := newNode
 }
```

حذف گره در لیست پیوندی

به‌طور مشابه توابعی برای حذف گره بعدی یک گره داده شده و همچنین گره ابتدایی فهرست وجود دارند. شکل نحوه حذف را نشان می‌دهد.



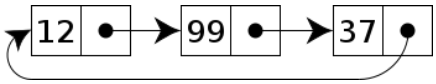

```
function removeAfter(Node node) // remove node past this one
    obsoleteNode := node.next
    node.next := node.next.next
    destroy obsoleteNode

function removeBeginning(List list) // remove first node
    obsoleteNode := list.firstNode
    list.firstNode := list.firstNode.next // point past deleted node
    destroy obsoleteNode
```

۳. انواع لیست پیوندی

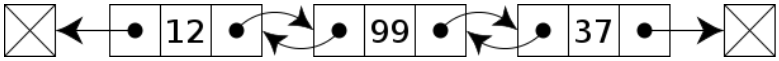
لیست پیوندی حلقوی

در آخرین گره یک لیست پیوندی ساده ، قسمت پیوند حاوی یک مرجع خالی است. یک قرارداد این است که آن را به اولین گره لیست ارجاع دهیم. در این صورت، گفته می شود که این لیست پیوندی "دایره ای" است. در غیر این صورت گفته می شود "خطی" است.



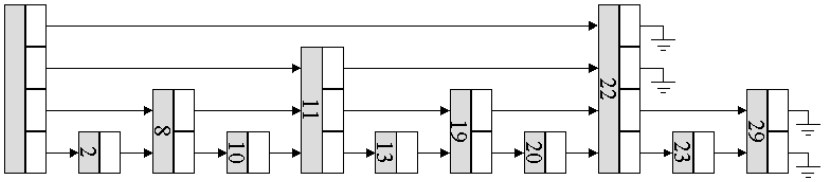
لیست پیوندی دو طرفه

در یک "لیست پیوندی مضاعف"، هر گره علاوه بر پیوند گره بعدی، یک قسمت پیوند دوم نیز دارد که به گره "قبلی" در دنباله اشاره دارد. این دو پیوند ممکن است "جلو" و "عقب" یا "بعدی" و "قبلی" نامیده شوند.



لیست های پرشی

در لیست‌های پیوندی معمولی هزینه جستجو، درج و حذف $O(n)$ است زیرا برای پیدا کردن گره داده مربوط، گره‌ها به ترتیب باید پیمایش شوند در این صورت اگر ما بتوانیم از روی بعضی گره‌ها بپریم می‌توانیم هزینه جستجو را پایین بیاوریم مثلاً به داده ساختار زیر دقت کنید :



در این شبه فهرست پرشی سایز هر دو گره متوالی ۲، سایز هر چهار گره متوالی ۳ و سایز هر 2^i گره متوالی $i+1$ است. همچنین سایز گره آغازین از همه بیشتر است.

شیوه جستجو در این شبه فهرست پرشی مانند جستجوی دودویی است یعنی برای پیدا کردن داده مورد نظر، هر بار با پایین آمدن تعداد اعداد نصف می‌شوند (ابتدا از بالاترین مرحله شروع می کنیم مثلاً اگر فهرست ما ۳۲ گره داشته باشد، ابتدا از گره شانزدهم شروع می کنیم و هر بار با جلو رفتن یا پایین آمدن، تعداد اعداد نصف می‌شوند)؛ واضح است که هزینه جستجو در این داده ساختار از مرتبه $O(\log n)$ است دقیقاً شبیه جستجوی دودویی.

اما تفاوت اصلی لیست های پرشی با این شبه لیست پرشی یا حتی لیست پرشی دودویی در تعداد اشاره‌گر های خروجی هر گره است، در واقع لیست پرشی یک لیست پیوندی است که هر گره آن می‌تواند به چند گره بعد از خودش اشاره کند که تعداد آن به صورت تصادفی مشخص می‌شود.

۴. تمارین مروری

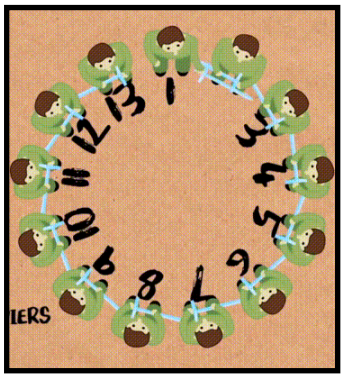
۱. با استفاده از چند آرایه یک لیست پیوندی خطی بسازید؛ با استفاده از چند لیست پیوندی یک آرایه بسازید. چه نتیجه ای می‌گیرید؟

۲. با استفاده از یک لیست پیوندی، یک صف بسازید؛ با استفاده از چند صف یک لیست پیوندی بسازید.(کمینه تعداد صف های لازم ؟)

۳. با استفاده از یک لیست پیوندی، یک پشته بسازید؛ با استفاده از چند پشته یک لیست پیوندی بسازید.(کمینه تعداد پشته های لازم ؟)

۴. با یک بار پیمایش لیست پیوندی، عنصر میانه لیست را پیدا کنید. (عنصر میانه را با میانگین اشتباه نگیرید)

۵. تمارین برنامه نویسی



۱. (ژوزفوس) (این همان سوال بیان شده در ۲-۱ است) افرادی را در نظر بگیرید که دایره وار ایستاده‌اند و منتظر اعدام هستند. بعد از آنکه اولین نفر اعدام می‌شود، تعداد مشخصی از افراد رد شده و یک نفر دیگر اعدام می‌شود. سپس دوباره به همان تعداد افراد پرش شده و نفر بعد کشته می‌شود. این فرایند حذف، دور دایره(که با برداشتن افراد کشته شده کوچک و کوچکتر می‌گردد) ادامه می یابد تا زمانی که تنها یک نفر باقی می‌ماند که آزاد می‌شود. مطلوب، یافتن جایگاهی در دایره اولیه است که شما با قرار گرفتن در آنجا نجات خواهید یافت.

پ.ن: به شدت پیشنهاد می‌شود که این پیاده سازی را با آرایه ها نیز تکرار کنید.

۲. HackerEarth, Reversed Linked List

۳. HackerEarth, Remove Friends

۶. برای مطالعه بیشتر

1. Shene, Ching-Kuang. "A comparative study of linked list sorting algorithms." 3C ON-LINE 3.2 (1996): 4-9.

2. Tsakalidis, Athanasios K. "Maintaining order in a generalized linked list." Acta informatica 21.1 (1984): 101-112.