گزارش تمرین سری پنجم

هدف ازین تمرین پیاده سازی درخت maxheap می باشد.

این درخت باید دارای یک سری توابع مشخص از قبیل push,pop باشد.

❋تابع constructor:

در این تابع ارایه ی arr که شامل وکتوری از پوینترها به نود های درخت می باشد را خالی از المان میکنیم.

❋تابع copy constructor:

در این تابع باید ابتدا باید المان های ارایه arr موجود مربوط به copy_object پاک شوند. سپس به ازای هر نود، ارایه ی جدیدی به طول ۱ ساخته شود و در ان پوینتر نود قرار گیرد. برای کامل کردن ارایه arr آدرس نود را باید به آن اضافه کنیم. دلیل اینکه مستقیما رفرنس هر نود را به ارایه arr اضافه نکردیم این است که در صورت انجام این عمل؛ چون تمام درایه های ارایه رفرنس به یک متغیر می شوند پس در هر بار اضافه شدن نود جدید به درخت تمام المان ها تغییر میکنند و مساوی عدد اخری که به درخت وارد شده می شوند. این همان روشی است که برای push نیز استفاده کرده ایم.

❋تابع destructor:

باید هر ارایه ای که new شده است delete بشود و ارایه arr نیز خالی از المان گردد.

❋تابع push

برای اضافه کردن هر نود به درخت از روش گفته شده در copy constructor استفاده شده است. متغیری به نام ind برابر سایز ارایه در نظر میگیریم. همانطور که می دانیم سایز ارایه یکی بیشتر از اندیس اخرین المان درون ارایه است و محاسبات گفته شده در رابطه با والد و فرزند چپ و راست برای زمانی صادق است که اندیس از یک شروع شده باشد نه صفر. مثلا برای اولین نود یعنی i=1 فرزند چپ در خانه ی 2*i=2 و فرزند راست در خانه ی 2*i+1=3 قرار دارد در حالیکه اگر i=0 بود،فرزند چپ باید در خانه ی صفر و فرزند راست در خانه ی اول قرار داشته باشد که میدانیم این نمایش درست نمی باشد. پس اندیس فرزند چپ ind-1 و اندیس فرزند راست ind+1-1 میگردد که به همین صورت برای خوانایی بیشتر در کد قرار گرفته و ساده نشده است.

اگر سایز ارایه بزرگتر از یک بود نیاز به مقایسه است در غیر اینصورت درخت دارای یک المان می باشد و همان بازگردانده میشود. هنگامی که نود به درخت اضافه می شود ابتدا باید با مقدار والد مقایسه شود که اگر از آن بزرگتر باشد، باید مقدار این دو نود با هم تعویض گردد. برای تعویض این دو مقدار دو راه وجود دارد یکی استفاده روش قبلی یعنی در نظر گرفتن یک متغیر میانی برای ذخیره ی مقدار نود ها و دیگری استفاده از swap از رویکرد STL می باشد که برای راحتی کار در ادامه معادل دو روش را ذکر کرده ام ولی در کد از swap استفاده شده است.

```
/* first approach to swap */
T* arr_m = arr[ind/2-1];
arr[ind/2-1] = arr[ind-1];
arr[ind-1] = arr_m;

/* second approach to swap using STL */
std::swap(arr[ind/2-1],arr[ind-1]);
```

این عمل مقایسه تا جایی انجام میشود که نود از مقدار والد خود کوچکتر باشد.

❋تابع pop

ابتدا باید اخرین المان ارایه را با اولین المان جا ه جا کرده و المان اخر را نیز حذف کنیم. سپس تابع popMax صدا زده میشود مکه در آن سعی در maxify کردن درخت داریم.

❋تابع popMax

ابتدا از راس درخت شروع میکنیم. در هر مرحله برای هر نود باید فرزندان ان را تشخیص دهیم تا فرایند حذف راحتتر صورت گیرد. اگر فرزندی نداشت که انتهای درخت است و باید از تابع خارج شود.(اگر فرزند چپ نداشته باشد فرزند راست نیز ندارد چون درخت همسطح پر می سود پس کافی است که وجود فرزند چپ را با اندیس ان چک کنیم) حالتی دیگر وجو دارد که فرزند راست نداشته باشد و فقط فرزند چپ داشته باشد که در این صورت مقدار والد با فرزند چپ مقایسه می شود و در صورت کوچکتر بودن تعویض میگردند. در حالتی که نود هر دو فرزند چپ و راست خود را داشته باشد باید در هر مرحله مقدار نود چپ و راست آن مقایسه شود و ماکزیمم ان ها با مقدار نود والد در صورت بزرگتر بودن جا به جا میگردد و دوباره تابع به صورت بازگشتی برای سطح پایینتر صدا زده میشود.

❋تابع operator+

این تابع ابتدا یک کپی از ابجکت کنونی می سازد و تغییرات push را روی ابجکت کپی شده اعمال میکند و ابجکت اصلی بدون تغییر باقی می ماند.

کلاس Student:

Id و average دو متغیر private هستند پس برای دسترسی به ان ها نیاز به توابع getter داریم.

در constructor کلاس فقط متغیر های ورودی را به متغیر های کلاس assign میکنیم.

نوشتن copy constructor و move constructor برای این کلاس لازم نیست چون متغیر دینامیکی وجود ندارد و compiler به درستی قادر به کپی و move ابجت ها می باشد اما برای کامل بودن کلاس نوشته شده است.

اپراتور = همانند copy constructor عمل میکند.

اپراتوری نیاز است که در صورت مشاهده >> عمل کند و اطلاعاتی که لازم است چاپ کند.

برای اینکه فرم template نوشته شده در کلاس apmaxheap درست کا رکند نیاز است که اپراتور هایی که استفاده شده به درستی در کلاس student تعریف شود. مانند اپراتور کوچکتر که نیاز است مقدار نود ها را با هم مقایسه کند.

پ.ن. : امتحان میانترم که همان تمرین ۵ می باشد تا سر مبحث STL بود اما در این تمرین از swap که مبحثی از STL که بسیار ساده است استفاده کردم. خواهش میکنم ازین مورد برای کسر نمره چشم پوشی کنید.