بسمه تعالی

درس: ساختمان های داده ها دانشگاه فردوسی مشهد مدرس: غیاثی شیرازی

تمرین HashTable:

در این تمرین شما می‌بایست بدنه‌ی توابع زیر را در فایل HashTable.h :

```
HashTable(int (*hashFunc) (K), float maxLoadingFactor=0.7, int initCapacity=7) {
    // Write your code here
}

virtual  void assign(K key, V value)
{
    // Write your code here
}

virtual  void remove(K key)
{
    // Write your code here
}

virtual  V& operator[] (K key)
{
    // Write your code here
}
```

ویژگی‌های این کلاس به این شرح است:

۱. mCapacity که ظرفیت HashTable را مشخص می‌کند.
۲. mSize که تعداد داده‌های درون HashTable را مشخص می‌کند.
۳. mTable که در واقع همان آرایه مربوط به HashTable است. جنس این آرایه pair است که اولین ورودی آن کلید داده و دومین مقدار آن خود مقدار داده است. (جهت آشنایی بیشتر با pair می‌توانید به این لینک مراجعه کنید: https://www.educative.io/edpresso/what-is-a-cpp-pair)
۴. mStateTable این آرایه هم به اندازه ظرفیت HashTable است و مشخص می‌کند که کدام خانه ها خالی و کدام‌ها پر هستند)

۵. (K key) (*hashFunc) که در حقیقت اشاره‌گری به یک تابع است که وظیفه آن hash کردن key است. این تابع ورودی از نوع template دارد که با K در اینجا مشخص شده است. این همان key است که با دادن آن به این تابع مقدار hash بازگردانده می‌شود و می‌توانید با گرفتن باقی‌مانده آن نسبت به ظرفیت، اندیس مورد نظر را در mTable بیابید.

۶. mMaxLoadingFactor که مشخص می‌کند با پر شدن چند درصد از HashTable یا همان mTable اندازه آرایه ها (mTable و mStateTable) باید بزرگتر شوند. طبیعی است که این عدد باید بین صفر و یک باشد. همچنین خود صفر و یک نیز نمی‌تواند باشد.

تابع یا همان constructor کلاس HashTable که پارامتر های زیر را دریافت می‌کند:

۱. (K) (*hashFunc) : که همان اشاره‌گر به تابع hash کردن است.

۲. mMaxLoadingFactor که به صورت پیش فرض مقدار ۰,۷ را دارد. (یعنی با پر شدن ۷۰ درصد از mCapacity آرایه یا همان ظرفیت بزرگتر می‌شود)

۳. initCapacity: این پارامتر مقدار اولیه HashTable را مشخص می‌کند. این مقدار به صورت پیش فرض ۷ است

تابع assign که با دریافت کلید و داده، آن را در مکان مناسب خود در mTable قرار می‌دهد. از طرفی مقدار متناظر با آن خانه در mStateTable نیز باید true شود، چرا که آن خانه پر شده است. توجه کنید که در صورت وجود کلید در جدول mSize تغییری نمی‌کند و کافی است مقدار متناظر با کلید را جایگزین کنید.

** نکته ای که شما باید در پیاده سازی این تابع در نظر داشته باشید تغییر اندازه آرایه است. هنگامی که mSize به درصد تعیین شده از ظرفیت رسید (mMaxLoadingFactor) باید اندازه آن تغییر کند. پس برای تغییر اندازه از دو برابر مقدار قبلی (mCapacity) شروع کرده و آن را بزرگ می‌کنیم تا به اولین عدد اول بعدی برسیم. مثلا اگر ظرفیت اولیه ۸ باشد، ظرفیت جدید باید ۱۷ شود. (عدد اول انتخاب می‌کنیم زیرا در ذخیره سازی و بازیابی بهینه تر است و احتمال روی هم قرار گرفتن کلید ها کمتر می‌شود). پس از تغییر اندازه فراموش نشود که مقادیر قبلی دوباره باید در آرایه بزرگتر اندیس گذاری شوند و در جای صحیح خود قرارگیرند تا HashTable درست شود.

** جهت تغییر اندازه آرایه ها: شما باید با new کردن آرایه، آرایه مورد نظر را در حافظه heap بسازید و پس از منتقل کردن داده های قدیم به آرایه‌های جدید، به کمک []delete آرایه قبلی که دیگر استفاده ندارد را حذف کنید تا حافظه اضافه گرفته نشود. پس از حذف کافی است فیلد ها به آرایه جدید که بزرگتر است اشاره کنند.

تابع remove کلید یک داده را گرفته و آن را از HashTable حذف می‌کند. لازم است که پس از حذف، مقادیر بعد از کلید حذف شده بررسی شوند و درصورت لزوم به جای خالی منتقل گردند. این کار تا زمانی که به خانه خالی برسیم ادامه دارد. (توجه کنید که HashTable هیچ گاه نباید کاملا پر شود. جهت مرور می‌توانید به اسلاید های HashTable مراجعه کنید)

در نهایت لازم است تا []operator را پیاده کنید. این عملگر مانند صدا زدن خانه ای در آرایه عمل می‌کند، با این تفاوت که بجای شماره خانه، در قلاب کلید داده مورد نظر را قرار می‌دهیم و عملگر reference ای از مقدار آن کلید را بازمی‌گرداند. مثلا hashTable[5] که مقدار مربوط به کلید ۵ (کلید از جنس int) را باز می‌گرداند. دقت کنید که این مقدار ممکن است object ای از یک نوع دیگر باشد که کلید آن عدد ۵ بوده است.

در این تمرین نحوه ارزیابی به شرح زیر است:

۱. testAssign: در این تست نحوه صحیح اضافه کردن داده به HashTable و همچنین نحوه صحیح تغییر اندازه و استفاده از mMaxLoadingFactor تست می‌شود. به دلیل اهمیت بالا این تست، ۳۵٪ از نمره این تمرین به آن اختصاص داده شده است. همچنین لازم به ذکر است که از آنجایی که این تست، تست پایه ای برای HashTable به حساب می‌آید، در صورت نمره نگرفتن از این تست از دو تست دیگر هم نمره‌ای دریافت نخواهید کرد. به عبارتی این تست پیش نیاز دو تست دیگر است.
۲. testDeleting: در این تست ابتدا مقادیری به HashTable اضافه شده و پس از آن در تعدادی از داده های موجود در آن حذف می‌شوند و در هربار حذف بررسی می‌شود که وضعیت HashTable صحیح بوده و داده ها به درستی جابجا شده اند. این کار بر روی چند HashTable انجام می‌شود. از آنجایی که شروطی که باید جهت جابجایی استفاده شود مهم است و لازم است که دانشجویان آن را به درستی یاد بگیرند به این تست نیز ۳۵٪ از نمره را اختصاص داده ایم.
۳. testHashFunc: در این تست ابتدا بررسی می‌کنیم که به درستی از تابع hash استفاده می‌شود و پس از آن پیاده سازی []operator بررسی می‌کنیم تا کلاس مربوط به کلید را به درستی بازگرداند. در نهایت بررسی می‌شود که از initCapacity درست انجام می‌شود و با تغییر آن، تغییر اندازه آرایه نیز همچنان صحیح عمل می‌کند. این تست ۳۰٪ از نمره را به خود اختصاص می‌دهد.

در نهایت توجه داشته باشید که پیاده سازی این تمرین پیچیدگی زیادی دارد و ممکن است نسبت به دیگر تمارین کمی دشوارتر باشد. پس زمان مناسبی جهت انجام آن درنظر گرفته شود.

برای انجام این تمرین کارهای زیر را انجام دهید:

۱- ابتدا در این پوشه فایل info.txt را با مشخصات خود پر کنید.
۲- سپس یک پروژه Visual Studio C++ بسازید و فایل های پوشه های src و test را به آن اضافه کنید.
۳- تمرین را انجام دهید و بخش‌های ناقص کد را تکمیل کنید و از درستی پیاده سازی خود مطمئن شوید.
۴- از پوشه src همه فایل های اضافی که به دلیل کامپایل برنامه بوجود آمده اند را پاک نمایید. (پوشه Debug و فایل با پسوند sdf و نیز پوشه مخفی vs را حتما پاک کنید زیرا حجم زیادی می گیرند).
۵- محتویات کل پوشه را به صورت یک فایل zip در آورید.
۶- مطمئن شوید که وقتی فایل zip را باز می کنید پوشه های src, img و test و همچنین فایل info.txt را می بینید.
۷- نام این فایل zip را به «شماره تمرین-شماره دانشجویی» تغییر دهید (مثل: HashTable.zip -123456789)
۸- دقت کنید که پسوند فایل شما حتما zip باشد.
۹- اگر حجم فایل بالای یک مگابایت باشد، حتما در پاک کردن محتویات بوجود آمده هنگام کامپایل (مرحله ۴) اشتباه کرده‌اید.
۱۰- ابتدا این فایل را به سیستم «سپهر» ایمیل کنید تا از نحوه عملکر برنامه خود بر روی تست های تکمیلی آگاه شوید.
۱۱- اشکالاتی را که سیستم «سپهر» مشخص کرده است برطرف نمایید و مجددا تمرین را به سیستم «سپهر» تحویل دهید.
۱۲- مرحله قبل را آن قدر ادامه دهید که از صحت عملکرد برنامه خود اطمینان حاصل نمایید.
۱۳- پس از اطمینان از دریافت نمره کامل، همان فایل ارسالی را از طریق سیستم vu تحویل دهید.

با آرزوی موفقیت