به نام خدا

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

نام اعضای گروه : مهسا آقایی و عاطفه علی محمدی

مستند پروژه درس ساختمان داده ها

استاد : دکتر فاطمی

توابع :

findTables

این تابع با دریافت نقشه ی رستوران و ابعاد آن تمامی میز های خالی را به ما برمیگرداند. این تابع از مرتبه O(n*n) است.

---

setTables

این تابع با دریافت فرد و اطلاعات فرد و لیست میزهای خالی و صف انتظار،چک میکند که اگر میز خالی وجود داشت به فرد میز میدهد و در غیر اینصورت فرد را در لیست انتظار قرار میدهد. این تابع از مرتبه O(1) است.

---

sortArr

این تابع لیست میز ها را میگیرد و مرتب میکند. این تابع از مرتبه O(n*n) است.

Definition

این تابع برای تعریف تعدادی نمونه از غذا ها و قرار دادن نمونه ها در لیست غذا ها استفاده میشود تا هنگام چاپ منو از آن استفاده شود.

هم چنین برای ست کردن تایم هر آماده شدن هر غذا به دلخواه زمانی در نظر گرفتیم که اینجا مشخص کرده ایم. این تابع از مرتبه O(1) است.

PrintExitPersons

این تابع افرادی که باید از رستوران خارج شوند را میگیرد که این افراد در یک ضف اولویت قرار دارند که بر اساس مجموع تایم آماده سازی غذا و تایم خوردن غذا این صف اولویت بندی شده است.

در این تابع هر فرد که از صف خارج میشود پیغام اتمام غذا و خروج فرد از رستوران چاپ میشود و سپس میز آن فرد به لیست میز های خالی اضافه میشود.

پس از آن اگر فردی در لیست انتظار قرار داشت با استفاده از تابع setTable به آن فرد یک میز داده میشود.و سپس آن را به صف اولویت راه ها که در ورودی دریافت کرده بودیم(printRoute) اضافه میکنیم تا بر اساس تایم تحویل غذا، غذای فرد تحویل داده شود و مسیر چاپ شود.(چاپ مسیر و تحویل غذا ها در تابع مین و با استفاده از یک وایل انجام شده است).

مرتبه ی زمانی: با توجه به اینکه حذف عنصر و اضافه کردن عنصر به صف اولویت از مرتبه ی O(logn) است و حذف و اضافه کردن داخل حلقه قرار دارند پس این تابع از مرتبه O(nlogn) است.

Convert

این تابع گراف را در یک اری لیست از اری لیست ها قرار میدهد . به این صورت عمل میکند که تمامی کاراکتر ها به جز کاراکتر وسایل تزیینی (#) را در گراف اد میکند یعنی رابطه ی هر گره را با گره دیگر در اری لیست مربوط به خودش اد میکند.پیچیدگی زمانی این تابع O(n*n)است.

BFS

این تابع شبیه به پیمایش اول سطح برای درخت عمل میکند. اما با این تفاوت که ما گراف ها ممکن است دارای چرخه باشند و ممکن است یک گره بیش از یکبار پیمایش شود و برای جلوگیری از این اتفاق از یک آرایه ی بازدید شده ی بولین استفاده میکنیم و هم چنین فرض میکنیم همه ی راس ها از راس شروع قابل دسترس باشند. مرتبه ی زمانی این تابع O(V+E)است.که در آن V تعداد راس ها و E تعداد یال هاست.

---

printShortestDistance

این تابع برای چاپ کوتاه ترین مسیر است که در این تابع از تابع BFSاستفاده کردیم که روند کار آن توضیح داده شد. مرتبه ی زمانی تابع نیز O(V+E)است.که در آن V تعداد راس ها و E تعداد یال هاست.

Insert

(مربوط به بخش مهمانی و در کلاس AVLTree)

این تابع برای اضافه کردن عنصر به درخت AVL است.(بخش مهمانی)

این تابع به این صورت عمل میکند که ابتدا عنصر را به درخت اضافه میکند (برای اضافه کردن ابتدا باید با استفاده از سرچ جای مناسب قرار گیری عنصر در درخت را تشخیص دهیم (عملکرد تابع **سرچ** به این صورت است که با توجه به اینکه هر گره از زیرشاخه ی چپ خود بزرگ تر یا با آن مساوی است و از زیر شاخه ی راست خود کوچک تر یا با آن مساوی است محل مناسب برای قرار گیری را تشخیص میدهیم و تابع سرچ از مرتبه O(logn) است) و سپس عنصر را در آن مکان قرار دهیم و اضافه میکنیم.)

متغیر balance اختلاف ارتفاع زیر شاخه ی چپ و راست نود ما را محاسبه میکند و هنگامی که بزرگ تر از 1 است و همچنین مقدار ما از مقدار node.right ما یک leftleft کیس داریم و باید یک عملیات leftRotate روی نود انجام دهیم.

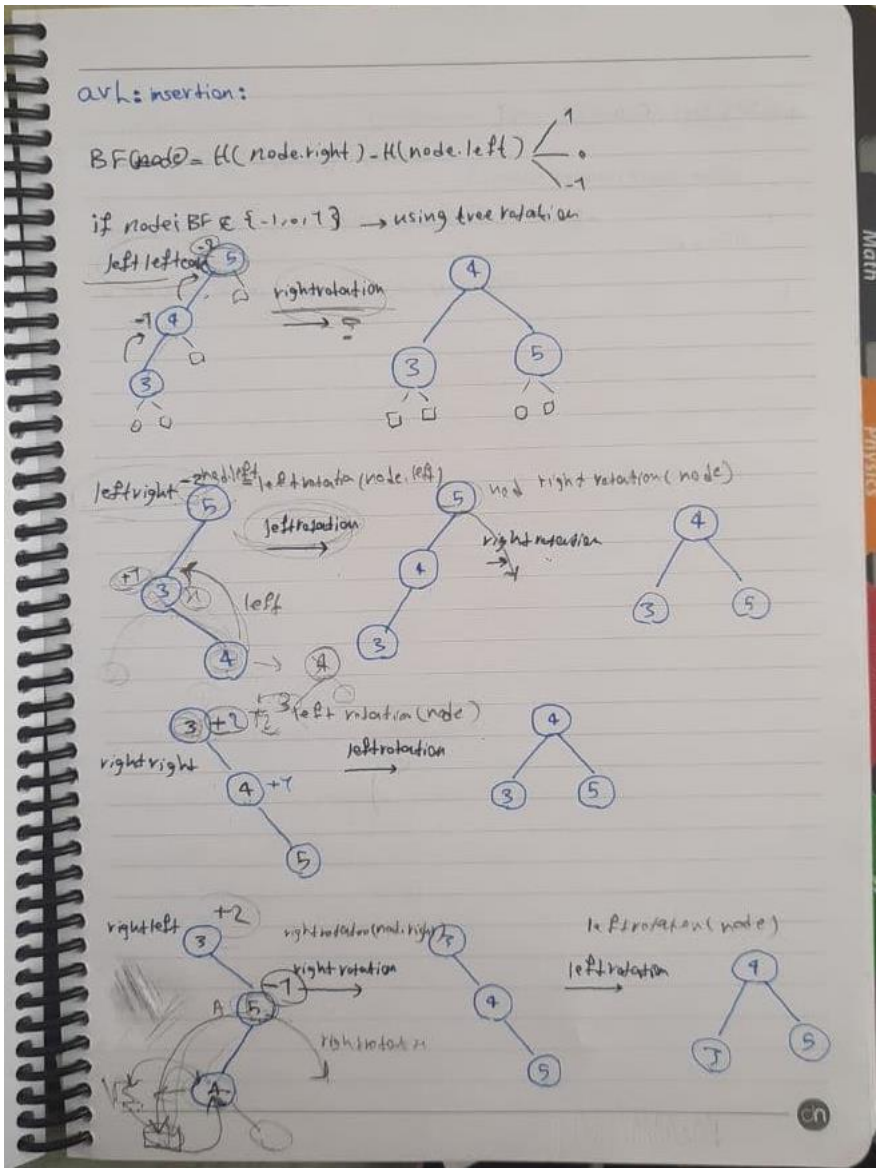
و هنگامی که balance کوچکتر از 1- است یعنی زیر شاخه ی راست و از چپ بلند تر و مقدار متغیر از مقدارnode.right بیشتر است یک کیس rightright داریم و باید یک عملیات leftRotate روی نود انجام دهیم.

هنگامی که balance بزرگتر از 1 یعنی زیرشاخه ی چپ از یک بزرگ تر ولی مقدار متغیر از node.left بیشتر است یک کیس leftright داریم پس ابتدا روی node.left یک leftRotate میزنیم و سپس روی نود یک leftRotate و برای rightleft نیز به همین صورت.

زمانی که balance کوچک تر از 1- است یعنی ارتفاع زیر شاخه ی راست بیشتر از زیر شاخه ی چپ است ولی مقدار کوچک تر از node.right است یک کیس rightleftداریم پس یک عمل rightRotate روی node.right میزنیم و سپس یک عملیات leftRotate روی نود اصلی میزنیم.

در این تابع از توابعی مانند بالانس و سرچ کمک گرفتیم که روند کار کامل توضیح داده شد.

در تصویر زیر روند کار با یک مثال شرح داده شده. این تابع از مرتبه O(logn) است.



Preorder

(مربوط به بخش مهمانی و در کلاس AVLTree)

این تابع برای نمایش درخت است که به صورت پیمایش preorder است. این تابع از مرتبه O(n) است.

Delete

(مربوط به بخش مهمانی و در کلاس AVLTree)

برای این تابع هم مانند تابع insert ابتدا به دنبال کلیدی که میخواهیم دیلیت کنیم میگردیم و اگر آن کلید موجود بود آن را حذف میکنیم و اگر نباشد null برمیگرداند. این تابع از مرتبه O(logn) است.

اگر موجود بود و حذف شد دوباره مانند تابع insert ارتفاع ها را چک میکنیم و اگر توازن درخت به هم خورده بود دوباره از تابع بالانس استفاده کنیم و درخت را بالانس میکنیم.

Calculator

(در کلاس Food و برای محاسبه ی تایم آماده سازی غذا)

این تابع برای محاسبه ی زمان آماده شدن یک غذا استفاده میشود.روند کار به این شکل است که نام یک غذا و آرایه ی تمامی غذا ها را در ورودی میگیرد و میگردد و نام غذا را در لیست غذا ها پیدا میکند و زمان آماده شدن غذا را که از قبل در تابع definition مشخص کرده بودیم را برمیگرداند. این تابع از مرتبه O(n) است.