مزایده گران عاقل :WiseBidders

هدف این پروژه، پیاده سازی پروتکل مزایده گری توضیح داد شده است. گرچه انجام اینکار با روشی ساده‌تر و بدون استفاده از اصول شی گرایی هم ممکن بود، اما برای اینکه کار را اصولی انجام دهیم و همچنین کد قابل فهم و قابل بسط یا دوباره استفاده کردن باشد، در این پروژه از اصول شی گرایی بهره جستیم.

از این رو، پس از دریافت ورودی‌های مذکور طبق صورت سوال، اعم از ماتریس حداکثر سرمایه خریداران (V) و لیست حداقل قیمت‌های خانه‌ها و ...، آن‌ها را طی فرایند preprocessing تحت قالب شی گرای طراحی شده می‌بریم و پس از آن فرایند مزایده و خرید و فروش را پیاده‌سازی می‌کنیم که الگوریتم آن بسیار ساده تر و قابل فهم تر از حالت کلاسیک خواهد بود.

۱. ساختار کلی و الگوریتم

پروژه شامل چند کلاس اصلی می‌باشد که طبق نامشان کاربردشان هم قابل پیش‌بینی است اما در اینجا باز به توضیح آن‌ها در بخش مربوطه می‌پردازیم.

الگوریتم کلی به این صورت است که پس از دریافت داده خام ورودی، اطلاعات حداقل قیمت خانه ها را به لیستی از عناصر کلاس House تبدیل می‌کنیم، همچنین داده‌های خریدارها را هم به همین منوال استخراج کرده و تحت قالب لیستی از عناصر کلاس Bidder ذخیره می‌کنیم.

سپس یک شی Auction خواهیم ساخت که وظیفه پروتکل مزایده را برعهده دارد. این کلاس با فراخوانی متد start خود، طبق اطلاعات House ها و Bidder ها و پیشنهادهای طبق پروتکلی تحت قالب دیکشنری ای از عناصر کلاس Bid ایجاد می‌کند و با استفاده از این عناصر هر راند مزایده را مدیریت می‌کند. نهایتا وقتی قیمت‌ها به سکون رسیدند خریدارهای نهایی و صاحبان نهایی خانه‌ها اعلام می‌شوند.

۲. کلاس‌ها

حال به تشریح کلاس ها بصورت مستقل می‌پردازیم:

۲-۱ کلاس House

این کلاس اطلاعات هر خانه داخل مزایده را نگه می‌دارد؛ اکرچه در این مورد داده اصلی صرفا حداکثر قیمت موردنظر فروشنده است، با این وجود ما دو متغیر دیگر با نام‌های id و name هم برای این کلاس جهت ایجاد تمایز میان خانه‌ها تعریف کرده‌ایم.

این کلاس تابع ممبر خاصی ندارد، صرفا یک مشخصه به نام is_soldکه نمایانگر فروش رفتن یا نرفته خانه است دارد. همچنین یک تابع استاتیک در این کلاس با نام ArrangeHouseInstances تعریف شده که وظیفه‌اش دریافت ورودی به فرم خواسته شده سوال (لیستی از حداقل قیمت خانه‌ها) و تبدیل آن به لیستی از اشیاء House می‌باشد.

همچنین این کلاس دو فیلد مهم با نام‌های best_bid و sold_to دارد. Best_bid در هردور مزایده مشخصه کننده بهترین پیشنهاد حال حاضر آن خانه و sold_to -که درآخر مزایده مقدار می‌گیرد- مشخص کننده آبجکت خریدار این خانه است.

۲-۲ کلاس Bidder

این کلاس همانطور که از نامش برمی‌آید، مشخص کننده یک مزایده‌گر یا همان خریدار است. همانند کلاس House این کلاس هم مشخصه‌های id و name را جهت متمایز کردن اشخاص داراست. فیلد دیگری با نام purchased دارد که در انتهای بازی و وقتی یک خریدار پیشنهاد نهایی‌اش ثبت شده و خریدش نهایی می‌شود مقدار می‌گیرد و برابر با آبجکت خانه‌ی خریداری شده توسط خریدار می‌گردد. همچنین فیلد max_invest_per_house نیز لیستی از حداکثر سرمایه هر خریدار به ازای خانه‌های موجود در مزایده است.

درواقع پس از فراخوانی متد ArrangeBidderInstances -که متدی مشابه با متد ArrangeHouseInstances می‌باشد- با دریافت ماتریس $V_{n\,x\,m}$ -که درایه $v_{i\,x\,j}$ نشانگر حداکثر سرمایه خریدار j ام برای خرید خانه i ام است- لیستی از خریدارها را تولید می‌کند. مشخصات هر خریدار در هر یک از ستون‌های این ماتریس وجود دارد که هر سطر i در این ستون، معادل حداکثر سرمایه‌ی خرید یک خریدار، برای خرید خانه i ام است. درواقع فیلد max_invest_per_house کلاس Bidder معادل با یک ستون خاص در این ماتریس است.

۳-۲ کلاس Bid

این کلاس نمایانگر یک پیشنهاد مزایده در پروتکل ماست. در واقع هر پیشنهاد یک مزایده‌گر برای یک خانه‌ی خاص تحت قالب این کلاس می‌باشد. فیلدهای اصلی این کلاس فیلد item که مشخص کننده خانه‌ای که پیشنهاد خریدش داده می‌شود و فیلد bidder که مشخص کننده خریداری که پیشنهاد را می‌دهد است، می‌باشند. همچنین فیلدهای دیگری هم هستند که به صورت زیر می‌باشند:

- max_investment: که مشخص کننده حداکثر هزینه‌ای که bidder این پیشنهاد حاضر است برای item اش بکند. این مقدار درواقع از لیست max_investment_per_house یک bidder مقدار دهی می‌شود.
- suggested_price: مشخص کننده هزینه پیشنهادی حال حاضر (درهردور مزایده) می باشد.
- profit: یکی از فیلدهای مهم این کلاس می‌باشد که در ادامه مبنای انتخاب پیشنهاد اصلی bidder ها خواهد بود. طبق فرمول ذکر شده در پروتکل این فیلد مشخصه کننده مقدار سود حال حاضر این پیشنهاد برای خریدار می‌باشد. این سود با محاسبه فاصله suggested_price از max_investment خریدار برای item انتخابی این bid بدست می‌آید.
- lose_count: مشخص کننده تعداد دفعات شکست پیشنهاد است. منظور از شکست مشخص کننده تعداد دفعاتی است که خریدار نتواند برای item پیشنهاد دهد، یعنی میزان پیشنهاد ها از سقف max_investment این bid عبور کرده باشد.
- :failed طبق تعریف پروتکل، هنگامی که یک خریدار روی خانه ای مزایده می‌کند، اگر به تعداد دو دور متوالی نتواند پیشنهادی بدهد، دیگر نمیتواند به آن خانه پیشنهادی دهد. فیلد failed دقیقا مشخص کننده همین وضعیت است که وقتی True باشد دیگر آیتم bid را فاقد اعتبار میکند و در الگوریتم باعث کنار گذاشته شدن آن می‌گردد.

همچنین متدهای اصلی این کلاس به شرح زیر می‌باشند:

- can_raise: این تابع با دریافت ورودی new_price از قیمت بعدی که در مزایده باید برای یکم item پیشنهاد داده شود مطلع شده و با مقایسه آن با max_investment تعیین می‌کند آیا bidder قادر به افزایش پقیمت پیشنهادی خود هست یا کم کم باید کنار بکشد!

- lose: این تابع زمانی که خریدار نتواند پیشنهاد دهد، یعنی درواقع زمانی که خروجی can_raise منفی باشد، فراخوانی می‌کند و lose_count را افزایش می‌دهد. هم‌چنین طبق تعریف صورت سوال، اگر مقدار این فیلد ۲ یا بیشتر شد، bid را failed تلقی خواهد کرد.
- raise_price: این تابع با دریافت قیمت هدف، قیمت پیشنهادی این bid را افزایش می‌دهد و سپس سود جدید را طبق تعرفه های جدید محاسبه می‌کند.
- temp_win: این تابع در هر راند برای پیشنهادی که مناسبترین تلقی می‌شود فراخوانی می‌شود و درواقع همان فرایند برنده شدن موقتی را پیاده سازی می‌کند. این تابع با به روز رسانی فیلد best_bid خانه هدف با خود آبجکت bid این کار را انجام می‌دهد.
- win: این تابع یک تابع نهایی تلقی می‌شود که پس از مشخص شدن وضعیت نهایی bid ها و به ثبات رسیدن قیمت ها، در راند آخر فراخوانی شده و برنده و خریدار نهایی هر خانه را مشخص می‌کند.

۴-۲ کلاس Auction

این کلاس، پیاده کننده پروتکل مزایده می‌باشد و درواقع کلاس اصلی و مدیر برنامه می‌باشد. پس از گرفتن ورودی کاربرها در ماژول app و تبدیل آنها به فرمت شی‌گرا، یک آبجکت Auction ساخته می‌شود که به آن لیست bidder ها و house ها و شرایط مزایده از قبیل گام مزایده € و تعداد ماکزیمم راند (درصورتی که برنامه در تعداد راندهای بیش از حد زیاد گیر کرد، این مقدار مانع هنگ کردن برنامه میشود) پاس داده شده و با فراخوانی متد start مزایده طبق آنچه که تعریف شده شروع و راندهای آن مدیریت میگرند. درادامه الگوریتم کار start شرح داده میشود.

روند کلی کار این تابع، به این صورت است که با استفاده از لیست‌های خریدارها و خانه ها (تحت عنوان bidders و items) ابتدا برای هر bidder تابع analyze خود را فراخوانی می‌کند؛ این تابع تمامی پیشنهادهای مزایده آبجکت های bid) را به ازای هر خریدار می‌سازد و سپس به ترتیب مقدار profit آن‌ها برای خریدارش مرتب می‌کند. سپس هر کدام از این لیست‌های مرتب شده‌ی جدید را تحت قالب یک آیتم دیکشنری با کلید از نوع Bidder و مقدار List[Bid]، برای هر خریدار در می‌آورد. درواقع در نهایت، این دیکشنری شامل تمامی پیشنهادهایی که یک خریدار می‌تواند برای هر خانه بدهد می‌باشد. مرتب شدن پیشنهادهای یک bidder بر حسب profit به این معناست که در هر دور مزایده، به ازای هر خریدار، عنصر اول bid هایش مشخص کننده بهترین پیشنهاد وی در آن دور می‌باشد.

نکته: درصورتی که یک خریدار در ابتدای لیست خود چند bid با مقدار سود یکسان داشته باشد، با فراخوانی تابع randomize_first_bid_by_profit_if_needed پیشنهاد اصلی خریدار بصورت تصادفی در می‌آید [طبق همان چیزی که در پروتکل خواسته شده است]. حالا به ازای هر خریدار عنصر اول لیست bid هایش انتخاب شده، و به یک دیکشنری دیگر تحت عنوان bids_per_item اضافه می‌شود. دیکشنری جدید برای هر خانه لیست bid های داده شده به آن در آن راند را شامل است. سپس لیست های موجود در این دیکشنری سپس بر حسب suggested_price بصورت نزولی مرتب می‌شوند که این یعنی گران‌ترین پیشنهاد برای یک خانه همان عنصر اول این لیست خواهد بود. همانند قبل درصورتی که چند پیشنهاد با مقدار suggested_price مساوی موجود باشد از طریق فراخوانی تابع randomize_first_bid_by_price_if_needed بهترین bid راند کنونی بصورت تصادفی انتخاب می‌گردد.

طبق این توزیع داده که در بالا شرح داده شد، در راند اول پیشنهادها برای هر خانه لیست می‌شوند، سپس این لیست از طریق فراخوانی accept_best_bid بهترین پیشنهاد برای هر خانه را ثبت می‌کند (از طریق فراخوانی تابع temp_win بهترین bid)

سپس در راندهای بعدی، برای هر آیتمی که بهترین bid دارد، درصورتی که خریداری دیگر دارای bid ای باشد که هنوز قادر به افزایش قیمت باشد (can_raise او برابر با True شود) با فراخوانی raise_price به میزان € نسبت به بهترین قیمت قبلی پیشنهاد جدیدی ثبت می‌کند (البته اگر این bid بهترین پیشنهاد انتخابی یک bidder باشد.) دوباره طبق روش مذکور، دیکشنری ها آپدیت می‌شوند و temp_win ها فراخوانی می‌گردد و پیشنهادها و best_bid ها آپدیت می‌شوند؛ این روند تا زمانی ادامه پیدا می‌کند که قیمت‌ها در ۳ راند اخر کاملا ثابت بمانند؛ در آن صورت در هر item با فراخوانی متد win فیلد best_bid خود خریدار برنده را اعلام می‌کند. تمامی این اطلاعات هم در هر راند مزایده در کنسول چاپ می‌شوند و کاربر را مطلع می‌کنند.

در ادامه نمونه خروجی برنامه را برای ورودی زیر قرار داده ایم:

```
def get_test_input():
    V = [ [20, 30, 50],
          [10, 50, 40,],
          [30, 10, 90,],
        ]

    r = [20, 10, 15,]
    epsilon = 1
```

نمونه ای از خروجی برنامه:

```
Round 2 :
        Bid on House #3 :
                 On House #3: 16 by Bidder #1
                 On House #3: 15 by Bidder #3
        House #3 for 16 to Bidder #1
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 3 :
        Bid on House #3 :
                 On House #3: 16 by Bidder #1
                 On House #3: 17 by Bidder #3
        House #3 for 17 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 4 :
        Bid on House #3 :
                 On House #3: 18 by Bidder #1
                 On House #3: 17 by Bidder #3
        House #3 for 18 to Bidder #1
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 5 :
        Bid on House #3 :
                 On House #3: 18 by Bidder #1
                 On House #3: 19 by Bidder #3
        House #3 for 19 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 6 :
        Bid on House #3 :
                 On House #3: 20 by Bidder #1
                 On House #3: 19 by Bidder #3
        House #3 for 20 to Bidder #1
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 7 :
        Bid on House #3 :
                 On House #3: 20 by Bidder #1
                 On House #3: 21 by Bidder #3
        House #3 for 21 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 8 :
        Bid on House #3 :
                 On House #3: 22 by Bidder #1
                 On House #3: 21 by Bidder #3
        House #3 for 22 to Bidder #1
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 9 :
        Bid on House #3 :
                 On House #3: 22 by Bidder #1
                 On House #3: 23 by Bidder #3
        House #3 for 23 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
```

```
Round 14 :
        Bid on House #3 :
                 On House #3: 28 by Bidder #1
                 On House #3: 27 by Bidder #3
        House #3 for 28 to Bidder #1
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 15 :
        Bid on House #3 :
                 On House #3: 28 by Bidder #1
                 On House #3: 29 by Bidder #3
        House #3 for 29 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 16 :
        Bid on House #3 :
                 On House #3: 30 by Bidder #1
                 On House #3: 29 by Bidder #3
        House #3 for 30 to Bidder #1
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 17 :
        Bid on House #3 :
                 On House #3: 30 by Bidder #1
                 On House #3: 31 by Bidder #3
        House #3 for 31 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 18 :
        Bid on House #3 :
                 On House #3: 30 by Bidder #1
                 On House #3: 31 by Bidder #3
        House #3 for 31 to Bidder #3
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
Round 19 :
        Bid on House #2 :
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #2
        Bid on House #3 :
                 On House #3: 31 by Bidder #3
        House #3 for 31 to Bidder #3
Round 20 :
        Bid on House #2 :
                 On House #2: 10 by Bidder #1
                 On House #2: 10 by Bidder #2
        House #2 for 10 to Bidder #1
        Bid on House #3 :
                 On House #3: 31 by Bidder #3
        House #3 for 31 to Bidder #3
Finally:
House #1 Sold to Bidder #3 @  20
House #2 Sold to Bidder #1 @  10
House #3 Sold to Bidder #3 @  31
```