هوش مصنوعی و سیستم‌های خبره

دکتر آرمین سلیمی بدر

پروژه پایانی درس – فاز اول

پاییز ۱۴۰۳

فهرست مطالب

مقدمه

وستروس؛ سرزمینی که در آن آتش و یخ همواره در کشاکشی ابدی به سر می‌برند. خاندان‌ها با جاه‌طلبی و عطش قدرت، برای فتح تخت آهنین به جان هم افتاده‌اند. و شما نیز رویایی دارید—رویای نشستن بر تخت و حکمرانی بر هفت اقلیم؛ جایی که تنها قوی‌ترین‌ها به آن دست می‌یابند. اما برای تحقق این آرزو به کسی نیاز دارید که بازی دسیسه‌ها را بهتر از هر کس دیگری بشناسد... کسی مثل لرد واریس، «عنکبوت» اسرارآمیز و استاد بزرگ زمزمه‌ها.

واریس مردی است که به هیچ خاندان وفادار نیست و بازی سیاست را برای اهداف خودش پیش می‌برد. او در سایه‌ها حرکت می‌کند، زمزمه‌ها را به گوش‌ها می‌رساند و هیچ‌گاه نیرویی را بیهوده هدر نمی‌دهد. با چشمانی که در هر گوشهٔ وستروس حضور دارند، واریس می‌تواند کلید موفقیت شما باشد. اما یک مشکل بزرگ وجود دارد: شما تنها کسی نیستید که به دنبال استفاده از قدرت واریس هستید. حریف شما نیز نقشه‌ای مشابه دارد و می‌خواهد از قدرت واریس برای رسیدن به اهدافش بهره ببرد.

در بازی «بازی تاج‌وتخت: دست پادشاه»، شما و رقیبتان با حرکت دادن واریس در دربار، به دنبال جذب شخصیت‌های کلیدی و قدرتمند هستید. هر حرکت می‌تواند پرچمی دیگر به قلمرو شما بیفزاید و شما را یک قدم به رویای پادشاهی نزدیک‌تر کند. واریس، بی‌طرف و مرموز، به هر دو طرف کمک می‌کند؛ او به شما ابزار لازم را می‌دهد تا به پیروزی برسید—اما هر لحظه ممکن است به دست حریف شما نیز بیفتد. در این نبرد، هر تصمیمی که بگیرید می‌تواند سرنوشت وستروس را رقم بزند.

این چالش شماست: آیا می‌توانید به کمک واریس، در بازی قدرت و سیاست، رقیب خود را شکست دهید؟ با هوش مصنوعی و الگوریتم *Minimax*، شما این فرصت را دارید تا عاملی برای هدایت واریس طراحی کنید؛ تا او را به سوی شخصیت‌ها بفرستید و شبکه‌ای از متحدان بسازید. آیا می‌توانید در دنیای بی‌رحم وستروس، به رویای خود برای حکمرانی جامهٔ عمل بپوشانید، پیش از آنکه حریف شما این فرصت را به دست بگیرد؟

آماده‌اید؟ اینجا وستروس است؛ جایی که هر حرکت و هر زمزمه می‌تواند شما را به تاج‌وتخت نزدیک‌تر یا از آن دورتر کند.

قوانین بازی: چگونه در کینگز لندینگ پیروز شویم؟

برای توسعه یک عامل هوشمند مؤثر، لازم است که ابتدا قوانین و اصول این بازی را به‌خوبی درک کنید. در اینجا، نحوهٔ کار بازی و نقش هر بازیکن توضیح داده شده است.

اجزای اصلی بازی: کارت‌ها و ابزارهای کلیدی

بازی برای دو بازیکن طراحی شده است. برای شروع بازی، موارد زیر را در نظر داشته باشید:

کارت‌های شخصیت خاندان‌های وستروس:

۳۵ کارت شخصیت که هر کدام نمایانگر یکی از اعضای خاندان‌های مشهور وستروس هستند، به‌صورت تصادفی در صفحه‌ای ۶×۶ چیده شده‌اند.هر کارت به یکی از خاندان‌های مختلف تعلق دارد که شما باید برای جذبشان تلاش کنید. در گوشه‌های کارت، رنگ و نشان خاص آن خاندان قرار گرفته است.

تصویر ۱ :یک نمونه از کارت های شخصیت، آریا استارک، که نماد خاندان او یعنی استارک در گوشه بالا سمت چپ

تعداد کارت‌های هر خاندان به شرح زیر است:

خاندان استارک: ۸ کارت

خاندان گریجوی: ۷ کارت

خاندان لنیستر: ۶ کارت

خاندان تارگرین: ۵ کارت

خاندان باراتیون: ۴ کارت

خاندان تایرل: ۳ کارت

خاندان تالی: ۲ کارت

کارت واریس

کارت واریس، نمایانگر لرد واریس است که در یکی از خانه‌های خالی صفحه قرار داده شده است. واریس ابزاری است که هر بازیکن می‌تواند از او برای رسیدن به اهداف خود بهره ببرد.

تصویر ۲ :کارت واریس

توکن‌های پرچم

توکن‌های پرچم، نمایانگر حمایت هر خاندان از بازیکنان هستند و هدف شما جمع‌آوری هر چه بیشتر این پرچم‌ها برای تسلط بر وستروس است.

نحوه بازی: حرکت دادن واریس و جمع‌آوری شخصیت‌ها

بازی به‌صورت نوبتی بین دو بازیکن پیش می‌رود. در هر نوبت، عامل شما باید واریس را در یکی از چهار جهت (بالا، پایین، چپ، راست) حرکت دهید. توجه داشته باشید که واریس نمی‌تواند اریب حرکت کند.

روند حرکت واریس و تغییرات ایجاد شده در کینگز لندینگ

- انتخاب هدف: بازیکن باید یک کارت موجود در چهار جهت اصلی خود را اعلام کند. پس از انتخاب، واریس به سمت آن کارت حرکت کرده و جانشین آن کارت میشود.
- جمع‌آوری کارت‌ها: با حرکت به سوی کارت مقصد توسط بازیکن، خود کارت شخصیت و تمامی کارت‌های شخصیت از خاندان آن کارت که در مسیر حرکت واریس قرار دارند، از صفحه بازی کینگز لندینگ جمع‌آوری شده و به عنوان دارایی شما ثبت میشوند.
- کسب پرچم: در پایان هر نوبت، در صورتی که بازیکن، تعداد کارت های مربوط به یکی از خاندان‌هایش مساوی یا بیشتر از حریف مقابل بود، پرچم خاندان مربوطه به او داده میشود که نشان‌دهنده تسلطتان بر شخصیت‌های آن خاندان هستند و جمع‌آوری آن‌ها شما را یک قدم به فتح وستروس نزدیک‌تر می‌کند!

نمونه‌ای از ابتدای بازی

برای درک بهتر قوانین و نحوه عملکرد بازی، بیایید یک نوبت نمونه را با جزئیات بررسی کنیم.

۱. وضعیت اولیه صفحه بازی (تصویر ۳)

تصویر ۳ :وضعیت اولیه صفحه بازی

واریس در گوشه بالا سمت چپ کینگز لندینگ قرار دارد و نوبت شماست.

- اگر عامل شما بخواهد واریس را به سمت پایین حرکت دهد، می‌تواند یکی از کارت شخصیت‌های *Hoster* از خاندان تالی، *Cersei* از لنیستر، *Daenerys* یا *Aegon I* از تارگرین، و یا *Robb* از استارک را به عنوان مقصد خود انتخاب کند.
- اگر عامل بخواهد واریس را به راست حرکت دهد، می‌تواند یکی از کارت شخصیت‌های *Viserys* از خاندان تارگرین، *Stannis* یا *Renly* از باراتیون، و یا *Victarion* از گریجوی را به عنوان مقصد خود انتخاب کند.
- اگر عامل بخواهد واریس را به چپ حرکت دهد، می‌تواند کارت شخصیت *Edmure* از خاندان تالی را به عنوان مقصد خود انتخاب کند.
- عامل نمی‌تواند واریس را به سمت بالا حرکت دهد، زیرا هیچ کارتی در سمت بالای واریس قرار ندارد.

۲. انتخاب هدف توسط عامل اول و حرکت به سوی آن (تصویر ۴)

تصویر ۴ :انتخاب هدف توسط عامل و حرکت به سوی آن

عامل هوشمند شما تصمیم می‌گیرد کارت *Renly* از خاندان باراتیون را به عنوان هدف خود انتخاب کند. واریس به سمت آن کارت حرکت کرده و جایگزینش می‌شود. با این کار، شما دو کارت شخصیت *Stannis* و *Renly* از خاندان

باراتیون را که در راه مسیر واریس بوده‌اند، تصاحب می‌کنید و این کارت‌ها از کینگز لندینگ خارج میشوند. در این لحظه سیستم بازی بررسی میکند که آیا تعداد کارت‌های خاندان باراتیون شما از حریف مساوی با، یا بیشتر از کارت‌های خاندان باراتیون رقیب است یا خیر، که در این مثال، شما ۲ کارت بیشتر از حریف دارید. پس پرچم خاندان باراتیون به شما داده میشود.

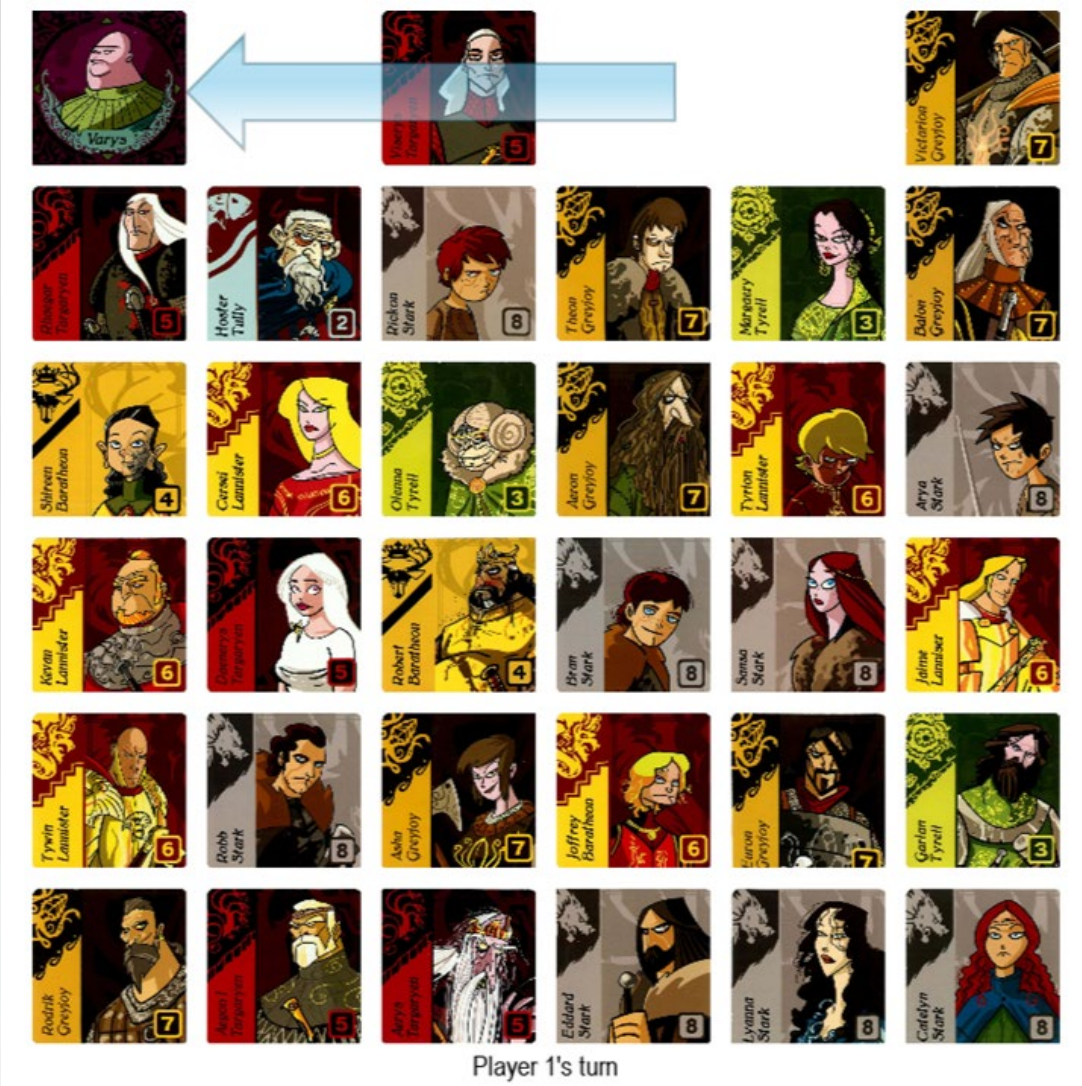
۳. وضعیت صفحه بازی پس از حرکت اول (تصویر ۵)

تصویر ۵ :وضعیت صفحه بازی پس از حرکت اول

واریس در گوشه بالا سمت راست کینگز لندینگ قرار دارد و نوبت حریف است.

- اگر عامل حریف بخواهد واریس را به سمت پایین حرکت دهد، می‌تواند یکی از کارت شخصیت‌های *Margaery* از خاندان تایرل، *Tyrion* از لنیستر، *Sansa* یا *Lyanna* از استارک، و یا *Euron* از گریجوی را به عنوان مقصد خود انتخاب کند.
- اگر عامل بخواهد واریس را به راست حرکت دهد، می‌تواند کارت شخصیت *Victarion* از خاندان گریجوی را به عنوان مقصد خود انتخاب کند.
- اگر عامل بخواهد واریس را به چپ حرکت دهد، می‌تواند یکی از کارت شخصیت‌های *Viserys* از خاندان تارگرین یا *Edmure* از خاندان تالی را به عنوان مقصد خود انتخاب کند.
- عامل نمی‌تواند واریس را به سمت بالا حرکت دهد، زیرا هیچ کارتی در سمت بالای واریس قرار ندارد.

٤. انتخاب هدف توسط عامل دوم و حرکت به سوی آن (تصویر ۶)



تصویر ۶ :انتخاب هدف توسط عامل دوم و حرکت به سوی آن

عامل هوشمند حریف تصمیم می‌گیرد کارت *Edmure* از خاندان تالی را به عنوان هدف خود انتخاب کند. واریس

به سمت آن کارت حرکت کرده و جایگزینش میشود. با این کار، حریف یک کارت شخصیت *Edmure* از خاندان تالی که در راه مسیر واریس بوده‌اند، تصاحب میکند و این کارت از کینگز لندینگ خارج میشود. در این لحظه سیستم بازی بررسی میکند که آیا تعداد کارت‌های خاندان تالی حریف از شما مساوی با، یا بیشتر از کارت‌های خاندان تالی شما است یا خیر، که در این مثال، حریف ۱ کارت بیشتر از شما دارد. پس پرچم خاندان تالی به حریف داده میشود.

پایان بازی و برنده شدن

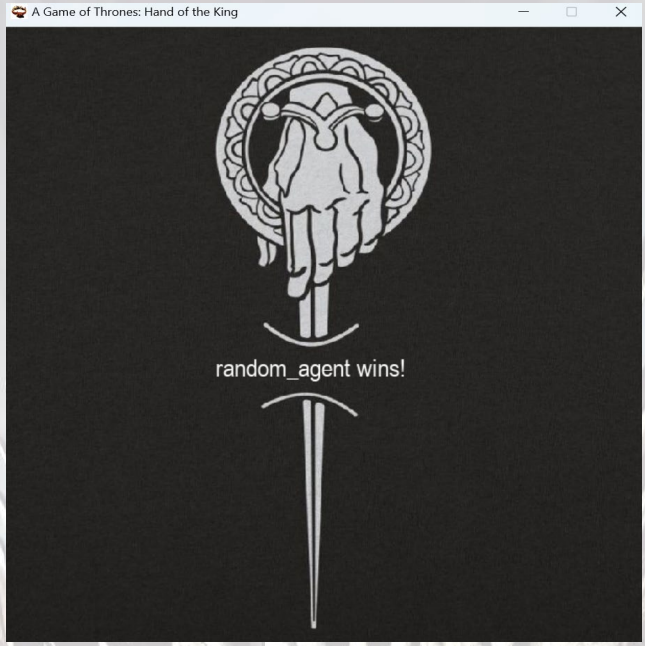
بازی زمانی به پایان می‌رسد که هیچ حرکت قانونی برای واریس باقی نمانده باشد. یعنی هیچ کارت شخصیتی در چهار جهت اصلی وجود نداشته باشد که واریس بتواند به سمت آن حرکت کند. در این شرایط، همه کارت‌های باقی مانده در کینگز لندینگ سوخته و بی ارزش میشوند. سپس بازیکنی که بیشترین تعداد پرچم‌ها را جمع‌آوری کرده باشد، به عنوان برنده بازی شناخته می‌شود. این بازیکن توانسته است به کمک واریس و متحدان خود، بیشترین نفوذ را در وستروس به دست آورده و به تاج و تخت را ازآن خود کرده است.

توجه: در صورتی که تعداد پرچم‌های فتح شده توسط دو بازیکن یکسان باشند، بازیکنی که پرچم خاندانی را در اختیار داشته باشد که بیشترین کارت شخصیت را دارد، برنده بازی خواهد بود.

مثال اول: فرض کنید در پایان بازی شما ۳ پرچم از خاندان‌های استارک، لنیستر و باراتیون را تصاحب کرده‌اید و حریف شما پرچم خاندان‌های گریجوی، تارگرین و تایرل را داراست. هر دو بازیکن ۳ پرچم را بدست آورده‌اند، در میان این پرچم‌ها، استارک با ۸ عضو (همانطور که در ابتدای توضیحات، و روی هر کارت نوشته شده است) بیشترین عضو را دارد و قدرت خاندانش بر سایرین ارجحیت دارد. پس سیستم بازی شما را به عنوان برنده بازی معرفی می‌کند!

مثال دوم: به تصویر زیر که از گزارش بازی در یکی از لحظات پایانی است، توجه کنید. در هر لحظه، تعداد کارت‌هایی که هر بازیکن از هر خاندان دارد چاپ می‌شود، و برای مشخص شدن اینکه چه کسی صاحب پرچم خاندان است، نام آن خاندان با رنگ سبز علامت گذاری میشود. تا این لحظه، بازیکن شماره دو به دلیل داشتن ۴ پرچم از خاندان‌های گریجوی، لنیستر، تارگرین و تالی برنده است. مشاهده میشود که تعداد کارت های خاندان تارگرین و لنیستر برای هر دو بازیکن یکسان است، اما چون بازیکن شماره دو، آخرین نفری است که کارت شماره چهارم را بدست آورده است، پرچم خاندان متعلق به اوست.

```
Player 1 cards status: Stark: 5 Greyjoy: 0 Lannister: 2 Targaryen: 2 Baratheon: 3 Tyrell: 2 Tully: 1
Player 2 cards status: Stark: 3 Greyjoy: 7 Lannister: 2 Targaryen: 2 Baratheon: 2 Tyrell: 0 Tully: 1
```



تصویر ۷ :صفحه پایان بازی و اعلام برنده

- پس عامل هوشمند شما، باید در زمان تصمیم گیری‌هایش به اهمیت هر خاندان نیز توجه کند.

راهنمای نصب پروژه

پیش‌نیازها

۱. داشتن نسخه مناسب پایتون:

مطمئن شوید که پایتون روی سیستم شما نصب است. برای بررسی، دستور زیر را در ترمینال یا خط فرمان اجرا کنید:

```
python --version
```

در صورتی که پایتون نصب نشده است یا نسخه‌ای قدیمی دارید، آخرین نسخه آن را از وب‌سایت رسمی پایتون دانلود و نصب کنید.

- پیشنهاد می‌شود از نسخه‌های جدید (مانند ۳٫۹ یا بالاتر) استفاده کنید.

۲. نصب کتابخانه‌های موردنیاز:

در پوشه پروژه، دستور زیر را برای نصب تمام کتابخانه‌های مورد نیاز اجرا کنید:

```
pip install pygame json argparse
```

- در صورتی که از محیط مجازی استفاده می‌کنید، اطمینان حاصل کنید که در همان محیط این دستورات اجرا شوند.

۳. اجرای پروژه:

پس از نصب پیش‌نیازها، برای اجرای بازی به فولدر پروژه بروید و فایل main.pyرا اجرا کنید:

```
python main.py
```

رفع خطاهای رایج

خطاهای مربوط به DLL (مانند libjpeg-9.dll)

اگر با خطاهای مربوط به فایل‌های DLL، مانند خطای زیر:

```
pygame.error: Failed loading libjpeg-9.dll: The file cannot be accessed by the system
```

مواجه شدید، به این ترتیب عمل کنید:

۱. بررسی مسیر نصب پایتون

به مسیر نصب پایتون خود بروید. مسیر نصب پایتون معمولاً در آدرس زیر میباشد:

```
C:\Users\<YourUsername>\AppData\Local\Programs\Python\Python<version>
```

بررسی کنید که فایل libjpeg-9.dll در پوشه‌های DLLs یا site-packages وجود دارد.

۲. راهکار جایگزین: بررسی مسیر نصب Pygame

با اجرای دستور زیر، مسیر نصب Pygame را پیدا کنید:

```
pip show pygame
```

سپس بررسی کنید که فایل libjpeg-9.dll در پوشه مربوط به Pygame وجود داشته باشد.

۳. دانلود فایل DLL در صورت عدم وجود

اگر فایل در مسیرهای بالا وجود ندارد، آن را از وب‌سایت‌های معتبر مانندdll-files.com دانلود کنید.فایل دانلودشده را به یکی از این مسیرها اضافه کنید:

٤. کپی فایل DLL به مسیر پروژه

اگر فایل را در مسیرهای بالا یافتید و همچنان در اجرای برنامه ارور قبلی را دریافت کردید، آن فایل را کپی کرده و فایل را در پوشه پروژه (مثل Hand-of-the-King)، در کنار فایل main.py قرار دهید.

بخش کدها: فایل‌ها و کلاس‌ها

برای درک نحوه عملکرد بازی و پیاده‌سازی عامل هوشمند، نیاز به آشنایی با ساختار فایل‌ها و اجزای اصلی کد داریم. این پروژه شامل فایل‌های زیر است:

۱. main.py:
فایل اصلی که حلقه بازی را مدیریت می‌کند. این فایل مسئول تنظیمات اولیه، اجرای منطق بازی، و تعامل با عامل‌ها و بازیکنان است.
۲. utils/classes.py:
شامل تعریف کلاس‌های اصلی بازی مانند کارت‌ها و بازیکنان است که داده‌ها و وضعیت بازی را مدیریت می‌کنند.
۳. utils/pygraphics.py:
ماژولی که نمایش گرافیکی بازی را بر عهده دارد. این بخش برای اجرای گرافیکی بازی ضروری است، اما نیاز به تغییر خاصی ندارد. این فایل به گونه‌ای طراحی شده است که اندازه صفحه و کارت‌ها بر اساس ابعاد نمایشگر شما تنظیم شود.

در ادامه، به توضیح مختصر کلاس‌های مهم موجود در فایل classes.py می‌پردازیم.

فایل classes.py

کلاس Card

نماینده هر کارت شخصیت در بازی، با اطلاعات زیر:

- خاندان (house): نام خاندان شخصیت.
- نام شخصیت (name): نام کاراکتر روی کارت.
- موقعیت (location): جایگاه کارت در صفحه بازی.

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23
24	25	26	27	28	29
30	31	32	33	34	35

برای مثال در تصویر بالا، برای چهار شخصیت اول، اطلاعات زیر را داریم:

```
cards = {list: 36} [<classes.Card object at 0x00000224FE8B1850>, <classes.Card object at 0x
  00 = {Card} <classes.Card object at 0x00000224FE8B1850>
    house = {str} 'No House'
    house_name = {str} 'No House'
    location = {int} 0
    name = {str} 'Varys'
  01 = {Card} <classes.Card object at 0x00000224FE8D2180>
    house = {str} 'Baratheon'
    house_name = {str} 'Baratheon'
    location = {int} 1
    name = {str} 'Robert'
  02 = {Card} <classes.Card object at 0x00000224FE8D21B0>
    house = {str} 'Targaryen'
    house_name = {str} 'Targaryen'
    location = {int} 2
    name = {str} 'Daenerys'
  03 = {Card} <classes.Card object at 0x00000224FE8D21E0>
    house = {str} 'Lannister'
    house_name = {str} 'Lannister'
    location = {int} 3
    name = {str} 'Tywin'
```

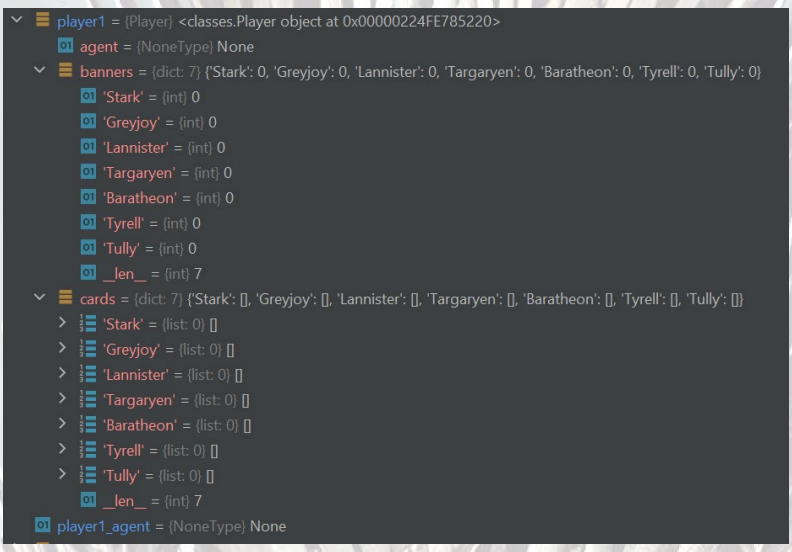
همانطور که گفته شد، واریس متعلق به هیچ خاندانی نیست.

کلاس Player

نماینده هر بازیکن در بازی، با ویژگی‌های زیر:

- کارت‌های تصاحب‌شده (cards): فرهنگ‌نامه‌ای از کارت‌های تصاحب‌شده برای هر خاندان.
- پرچم‌ها (banners): پرچم‌های تصاحب‌شده توسط بازیکن.

برای مثال در ابتدای بازی داریم:



فایل main.py

این فایل مسئولیت اجرای کلی بازی را بر عهده دارد. از تنظیمات اولیه تا مدیریت نوبت‌ها، پیاده‌سازی قوانین بازی و ارتباط با عامل‌ها (چه انسانی و چه هوش مصنوعی). توجه داشته باشید بازی به گونه ای پیاده سازی شده است که شما خود نیز میتوانید با عامل توسعه داده‌تان یا با دوستان خود بازی کنید. به طور خلاصه بازی در این سه حالت قابل اجرا میباشد:.

- Human VS Human
- Human VS agent
- Agent VS Agent

در این بخش، مهم‌ترین اجزای این فایل که برای استفاده از کد و توسعه عامل شما نیاز است توضیح داده می‌شوند:

پارامترهای ورودی از طریق آرگومان‌ها

در این فایل از کتابخانه argparse برای دریافت ورودی‌های اولیه استفاده می‌شود. این ورودی‌ها شامل موارد زیر هستند:

- --player1 و --player2: این پارامترها تعیین می‌کنند که بازیکن اول و دوم چه کسانی هستند:
  - مقدار human نشان‌دهنده بازیکن انسانی است. توجه داشته باشید که این حالت به صورت دیفالت ست شده است. پس اگر شما فقط به اجرای فایل main.py بپردازید، هر دو بازیکن انسان در نظر گرفته شده و میتوانید با کلیک کردن روی خانه‌ها به صورت نوبتی بازی کنید.
    - این دو حالت زیر برای اجرای بازی یکسان بوده و در آن‌ها بازیکنانش شما هستید.

```
python main.py
```

```
python main.py --player1 human --player2 human
```

  - اگر یک فایل کد (مثل my_agent.py) ارائه شود، به‌عنوان عامل هوشمند آن بازیکن در نظر گرفته می‌شود. پس برای اینکه عامل شما به عنوان یکی از بازیکن‌ها بازی کند، نام فایل را، بدون پسوند .py بنویسید.
    - در این حالت زیر بازیکن اول، عاملی است که کد آن در فایل minimax_agent.py نوشته شده است. بازیکن دوم نیز عاملی است که کد آن در فایل random_agent.py نوشته شده است.

```
python main.py --player1 "minimax_agent" --player2 "random_agent"
```

- --load یا --l : از این پارامتر برای بارگذاری وضعیت اولیه بازی از یک فایل *.json استفاده می‌شود. این قابلیت برای تست و اجرای دوباره بازی بسیار کاربردی است.
- --save یا --s : امکان ذخیره وضعیت فعلی بازی در یک فایل را فراهم می‌کند تا بتوان آن را برای تحلیل‌های بعدی استفاده کرد.
  - پس از ذخیره سازی یک صفحه در فایل *.json در پوشه boards می‌توانید آن را اینگونه بارگذاری کنید:

```
python main.py --load "screenshot"
```

عامل‌ها (Agents)

همانطور که گفته شد، بازی از بازیکنان انسانی و عامل‌های هوشمند پشتیبانی می‌کند:

- عامل انسانی: کاربر واریس را با کلیک کردن روی کارت‌ها حرکت میدهد.
- عامل هوشمند: فایل ارائه‌شده به‌عنوان عامل که با الگوریتم خود، واریس را حرکت میدهد.

برای توسعه عامل خود، ابتدا باید توجه داشته باشید که در فایل main.py در حلقه اصلی تابع main(arg) با این کد

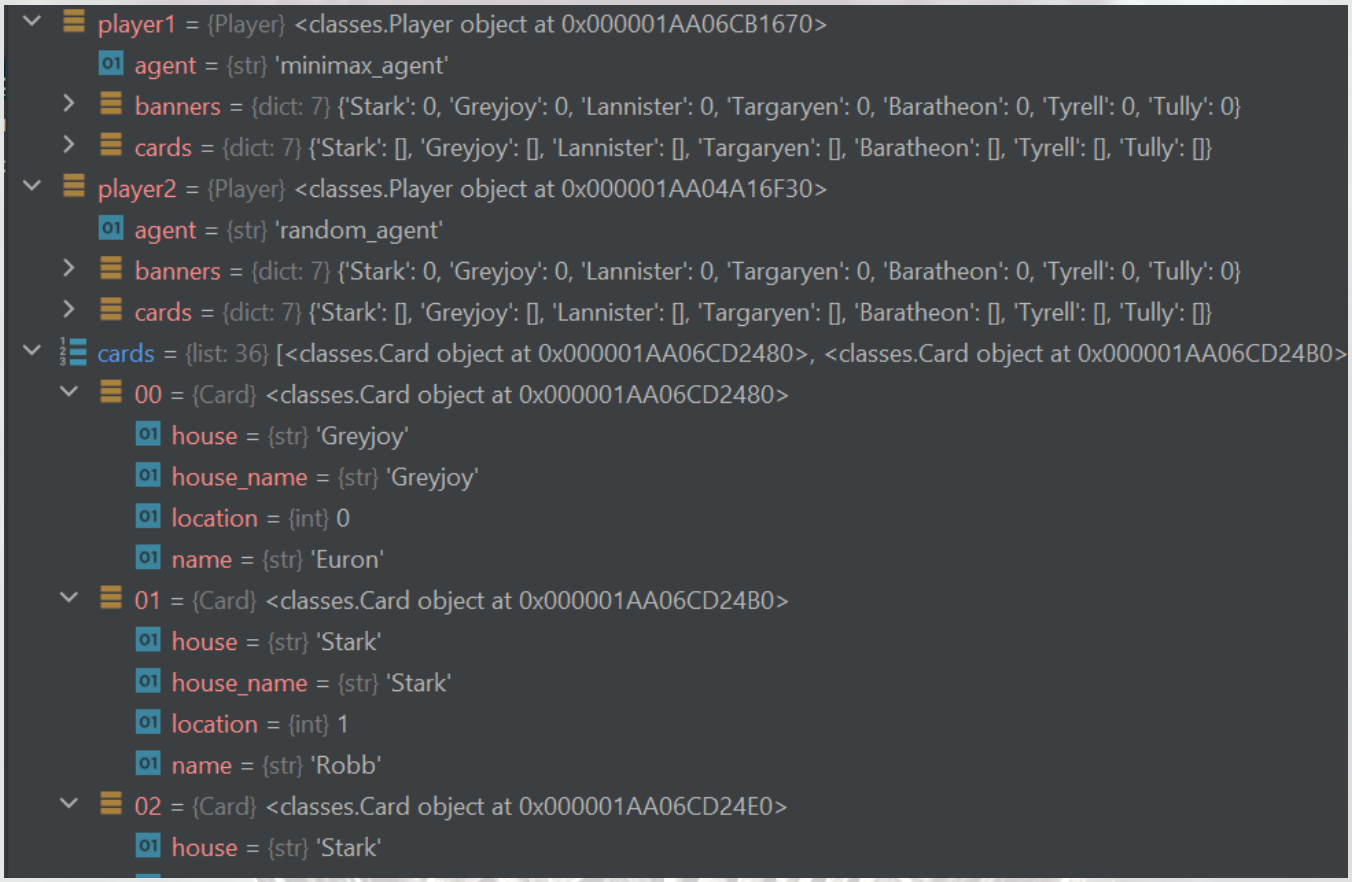
```
move = try_get_move(player1_agent, cards, player1, player2)
```

به سراغ فراخوانی تابع try_get_move میرود:

```python
def try_get_move(agent, cards, player1, player2):
    """
    This function tries to get the move from the AI agent.

    Parameters:
        agent (module): AI agent
        cards (list): list of Card objects
        player1 (Player): player 1
        player2 (Player): player 2

    Returns:
        move (int): location of the card
    """

    # Try to get the move from the AI agent in TIMEOUT seconds
    with concurrent.futures.ThreadPoolExecutor() as executor:
        future = executor.submit(agent.get_move, copy.deepcopy(cards), copy.deepcopy(player1), copy.deepcopy(player2))

        try:
            move = future.result(timeout=TIMEOUT)

        except concurrent.futures.TimeoutError:
            move = None

    return move
```

این تابع وظیفه دارد تا حرکت پیشنهادی عامل را دریافت کند. این کار از طریق متد get_move در فایل عاملی که شما پیاده سازی کرده‌اید انجام می‌شود. عملکرد آن به این شکل است:

۱. ورودی (مشاهدات عامل):
   - cards: آرایه‌ای از تمام کارت‌های موجود در صفحه، شامل اطلاعات هر کارت.
   - player1 و player2: وضعیت هر دو بازیکن (از جمله کارت‌ها و پرچم‌هایشان).
   - agent: عامل هوشمند ارائه‌شده.



۲. خروجی (کنش عامل):

عامل شما باید یک عدد صحیح (بین ۰ تا ۳۵) برگرداند که نشان‌دهنده کارت انتخابی برای حرکت واریس است.

۳. مدت‌زمان مجاز (TIMEOUT):

اگر عامل نتواند در مدت تعیین‌شده حرکت خود را اعلام کند، مقدار None بازگشت داده شده و نوبت بازیکن دیگر می‌شود.

نکته:

با توجه به توضیحات داده شده، عامل شما باید حتماً یک متد اصلی get_move را پیاده‌سازی کند. این متد به عنوان یک پلی بین کدهای بازی و الگوریتم‌های پیاده سازی شده شما میباشد که ورودی‌ها را دریافت میکند. پس از پردازش مشاهدات و اجرای الگوریتم minimax خود یک حرکت معتبر را بازگرداند.

دریافت پروژه:

پروژه را از لینک زیر دانلود کنید. ساختار پروژه به صورت زیر میباشد:

- assets
- boards
- utils
    - classes.py
    - pygraphics.py
- main.py
- random_agent.py

نحوه نمره‌دهی پروژه:

در این فاز هدف اصلی ما آشنایی شما با محیط بازی و الگوریتم minimax میباشد که بتوانید عاملی را بسازید، که با مشاهدات خود از محیط، کنش‌های مناسبی داشته باشد. بنابراین بخش اصلی نمره این فاز به پیاده‌سازی درست الگوریتم و استفاده هوشمندانه از آن در محیط خود، و ارائه با تسلط کافی میباشد. در مراحل اولیه، پروژه سعی کنید که با استفاده از عامل random_agent.py به رقابت با آن پرداخته و آن را شکست دهید. همچنین یک عامل minimax توسط تیم حل تمرین طراحی شده است، که زمان ارائه، بخش کوچکی از نمره شما به شکست دادن این عامل داده میشود. توجه داشته باشید که برای آسان کردن سطح رقابت، در زمان ارائه، عمق minimax عامل اشاره شده، روی مقدار کمی تنظیم میشود تا عامل هوشمند شما که به درستی طراحی شده، بتواند اون را شکست دهد.

نحوه ارسال و ددلاین پروژه:

پس از پیاده‌سازی توضیحات مناسبی از نحوه پیاده‌سازی را در قالب یک فایل PDF، به همراه کدهای پروژه خود، به صورت یک فایل زیپ شده در درس افزار آپلود کنید. در زمانی که متعاقباً اعلام خواهد شد، هر تیمی به ارائه مجازی این فاز از پروژه خود خواهد پرداخت. ددلاین این پروژه تا ۳۰ آذر ماه میباشد. با توجه به فاز بعدی پروژه، تمدید آن ممکن نخواهد بود.

قالب تیم‌ها:

فاز اول و دوم این پروژه می‌تواند در قالب تیم های یک‌نفره یا دونفره باشد. توجه کنید که تیم های انتخابی شما در هر دو فاز یکسان خواهد بود. به زودی محیطی را برای وارد کردن اعضای تیم‌های پروژه در گروه درسی ارسال میکنیم.

پاسخ به سوالات و رفع اشکال:

در صورت بروز هرگونه مشکل، سوالات خود را ترجیحا در گروه درسی بپرسید. در صورت نیاز می‌توانید با آی‌دی‌های لینک شده اعضای حل تمرین نیز در ارتباط باشید.

محمد متین مومنی | *صدرا برزگی* | *ماهان ویسی* | *غزال لقایی* | *نگار هنرور*


با آرزوی موفقیت و سربلندی برا شما 😊