۱ توضیحات کلی

در این پروژه شما باید بازیکنانی طراحی و پیاده‌سازی کنید تا در مصاف با بازیکنان دیگر دانشجویان درس به رقابت بپردازند. مسابقات بین بازیکنان در چهار سناریوی متفاوت تنظیم و برگزار می‌شود و نمره شما از این پروژه بر اساس امتیازاتی است که بازیکنان شما از این سناریوها دریافت می‌کنند. تمامی سناریوها بر پایه بازی اتمام حجت (Ultimatum Game) می‌باشند و در جزئیات تفاوت دارند. شما باید برای هر سناریو یک کد مجزا بارگذاری کنید. استراتژی‌های بازیکنان شما می‌توانند مشابه یا کاملا متفاوت باشند، همچنین می‌توانند به هر اندازه پیچیده باشند و حتی می‌توانید از مدل‌های ساده هوش مصنوعی نیز استفاده کنید! تنها قیود موجود، همخوانی کدهای شما با API سناریوی مربوطه، رعایت قیود زمانی، و استفاده از کتابخانه‌های معمول است. توجه داشته باشید که **این پروژه به شکل انفرادی انجام می‌شود**. در ادامه توضیحات دقیق‌تر درباره ابعاد مختلف پروژه آورده شده است.

۲ بازی اتمام حجت

تمامی سناریوهای پروژه بر اساس این بازی تعریف شده‌اند. در این بازی دو بازیکن وجود دارند. بازی به این شکل است که در آغاز بازیکن اول (میزبان) s امتیاز دریافت می‌کند و باید آن را بین خود و بازیکن دوم (مهمان) تقسیم کند. تقسیم به این صورت است که بازیکن اول تصمیم می‌گیرد چه مقدار از این s امتیاز را به بازیکن دوم بدهد و این مقدار را به او پیشنهاد می‌کند. حال بازیکن دوم دو انتخاب دارد: می‌تواند این پیشنهاد را قبول و یا رد کند. در صورتی که این پیشنهاد را قبول کند، به اندازه مقدار پیشنهادی امتیاز می‌گیرد و میزبان نیز باقی‌مانده امتیاز را برای خود برمی‌دارد. در صورتی که پینشهاد را رد کند، به هیچ یک امتیازی داده نمی‌شود و بازی تمام می‌شود. به صورت دقیق‌تر، اگر این بازی را به شکل $G_s = (A^1, A^2, u^1, u^2)$ نشان دهیم که A^1, A^2 مجموعه کنش‌های بازیکنان و u^1, u^2 توابع پاداش آن‌ها باشند و (a^1, a^2) یک پروفایل استراتژی باشد، داریم:

$$A^1 = \{r | r \in [0, s]\}, \quad A^2 = \{0, 1\}$$

$$u^1(a^1, a^2) = \begin{cases} 0 & a^2 = 0 \\ s - a^1 & a^2 = 1 \end{cases}$$

$$u^2(a^1, a^2) = \begin{cases} 0 & a^2 = 0 \\ a^1 & a^2 = 1 \end{cases}.$$

برای اطلاعات بیشتر به اینجا مراجعه کنید.

۳ سناریوهای بازی

پروژه در سه سناریو برگزار می‌شود که کلیت سناریوها مشابه و بر اساس بازی اتمام حجت است. تمامی سناریوها شامل تعدادی پارامتر هستند که بسته به سناریو تغییر می‌کنند. این پارامترها عبارتند از **حداکثر زمان مورد قبول در هر دست برای پاسخگویی، و اطلاعاتی که در هر دست در اختیار بازیکنان قرار می‌گیرد.** همچنین در صورت وجود پارامتری دیگر، در توضیحات سناریو مربوطه ذکر خواهد شد. در تمامی سناریوها امتیاز یک دست بازی (مقدار s) برابر 1 می‌باشد. در انتها هر دانشجو تعدادی امتیاز خواهد داشت (یک امتیاز برای هر سناریو) که امتیاز و نمره نهایی وی با توجه به آنها محاسبه خواهد شد. در نهایت، توجه داشته باشید که **درصورتی که کد شما بیش از حداکثر زمان مورد انتظار طول بکشد، در صورتی که نفر اول باشید پیشنهاد شما مقدار s و اگر نفر دوم باشید کنش شما قبول پیشنهاد ثبت خواهد شد.** برای کسب اطلاعات بیشتر در رابطه با نحوه و تفاوت پیاده‌سازی هر سناریو، به بخش ۴ مراجعه کنید.

۱.۳ سناریو اول: بازی مکرر کلاسیک

در این سناریو بین هر دو بازیکن یک بازی مکرر (Repeated Game) انجام می‌شود. به عبارتی دیگر، هر دو بازیکن 100 بار به صورت متوالی بازی می‌کنند و در نهایت جمع امتیازات کسب‌کرده از این 100 دست به امتیاز کسب‌کرده آن‌ها از این سناریو اضافه خواهد شد. منطق جفت شدن بازیکنان به صورت تصادفی است. میزان امتیاز هر دست یک و حداکثر زمان آن یک دهم ثانیه می‌باشد. در هر دست جایگاه مهمان و میزبان عوض می‌شود (هر بازیکن 50 بار مهمان و 50 بار میزبان خواهد بود). در این سناریو تنها اطلاعات مورد نیاز جهت اجرای بازی به بازیکنان داده می‌شود و بازیکنان موظفند هر داده اضافه‌ای را خود ذخیره و پردازش کنند.

۲.۳ سناریو دوم: بازی مکرر مخدوش

این سناریو مشابه سناریو اول است با این تفاوت که مقداری نویز به بازی‌ها اضافه می‌شود. در واقع در هر دست بازی، به مقداری که نفر اول پیشنهاد می‌دهد یک نویز تصادفی نرمال با میانگین صفر و واریانس 0.05 اضافه می‌شود و این مقدار به نفر دوم گزارش می‌شود. همچنین به احتمال 0.2 کنشی که نفر دوم گزارش می‌کند برعکس در نظر گرفته خواهد شد (اگر کنش انتخابی صفر بود، یک می‌شود و برعکس). مابقی موارد مانند سناریو اول است.

۳.۳ سناریو سوم: لالیگیم (لیگ)

در این سناریو، هر دو بازیکن یک بازی رفت و یک بازی برگشت انجام می‌دهند که هر بازی دو دست است (باری مهمان و باری میزبان). در این سناریو در آغاز هر و در کنار اطلاعات دیگر، کل تاریخچه بازی‌های بازیکن مقابل به شما داده می‌شود. این تاریخچه شامل شماره بازی، مهمان یا میزبان بودن بازیکن، کنش‌های انجام شده و نتایج آن‌هاست. امتیاز هر دست در این بازی 2 و حداکثر مدت پاسخگویی نیم ثانیه می‌باشد. با توجه به اینکه لیگ در 150 هفته برگزار می‌شود (تعداد تقریبی دانشجویان)، اگر دو بازیکن بازی اولشان را در هفته i ام انجام دهند، بازی دوم خود را در هفته $150 - i$ ام انجام می‌دهند.
توجه. در این سناریو برای عاملی که ارائه می‌کنید اسم و شعار انتخاب کنید!

۴ زمان و نحوه تحویل و موارد پیاده‌سازی

یک پوشه شامل تعدادی فایل پایتون در اختیار شما قرار می‌گیرد که شما باید بعضی قسمت‌های این فایل‌ها را پیاده‌سازی و سپس بارگذاری کنید. این فایل‌ها شامل موارد زیر هستند:

فایل player_phase1.py شامل کلاس مورد نیاز برای پیاده‌سازی سناریو اول و **فایل player_phase2.py** شامل کلاس مورد نیاز برای پیاده‌سازی سناریو دوم است. در هر دوی این فایل‌ها شما باید کلاس UGPlayerPhase1_STUNUM و UGPlayerPhase2_STUNUM را که STUNUM شماره دانشجویی خودتان است، پیاده‌سازی کنید؛ یعنی باید انتهای اسم این کلاس شماره دانشجویی خودتان را اضافه کنید. در هر کدام از این کلاس‌ها باید ۳ متد را پیاده‌سازی کنید:

- **reset()**: این متد تنها یک بار قبل از بازی با یک نفر صدا زده می‌شود. در این متد می‌توانید متغیرهای مورد نیاز خودتان را مقداردهی کنید.
- **proposer_strategy(round_number)**: در مراحلی که شما میزبان هستید، این متد از کلاس شما صدا زده می‌شود. round_number نشان‌دهنده این است که چندمین مرحله از ۱۰۰ بازی شما است و همیشه عددی بین ۱ تا ۱۰۰ است. خروجی این متد باید عددی صحیح بین ۰ تا ۱۰۰ باشد، که نشان می‌دهد چند درصد از امتیاز را به مهمان پیشنهاد می‌دهید.
- **responder_strategy(round_number, offer)**: در مراحی که شما مهمان هستید، شماره مرحله و پیشنهاد میزبان در ورودی این متد به شما داده می‌شود؛ در خروجی یک boolean خروجی می‌دهید که نشان می‌دهد پیشنهاد را قبول می‌کنید یا نه. خروجی True به معنی قبول کردن پیشنهاد و False به معنی نپذیرفتن آن است.

دقت کنید هر متدی از برنامه شما تنها ۱.۰ ثانیه مهلت دارد خروجی خود را بدهد. در صورتی که متد شما در زمان گفته شده، به پایان نرسد، یا با فرمت گفته شده مطابقت نداشته باشد؛ خروجی پیش‌فرض برای آن در نظر گرفته می‌شود. خروجی پیش‌فرض برای proposer_strategy s است و برای responder_strategy True است.
فایل **tester.py** نیز برای تست کردن برنامه خودتان است. دقت کنید که این فایل نهایی داوری نیست. برای تست کردن، فایل‌های خود را در بالای برنامه import کنید و برنامه را اجرا کنید.
فایل agent.py را باید برای سناریو ۳ کامل کنید. این فایل نیز مشابه دو فایل قبلی، یک کلاس دارد که باید شماره دانشجویی خود را به انتهای آن اضافه کنید. در ابتدای هر بازی، یک نمونه از کلاستان ایجاد می‌شود و در آن تاریخچه بازی‌های خودتان و بازیکن مقابلتان داده می‌شود. بعد از آن، اگر در آن هفته میزبان باشید، متد proposer_strategy فراخوانی می‌شود و در صورتی که مهمان باشید، متد responder_strategy فراخوانی می‌شود. به کمک فایل agent-tester.py می‌توانید یک داده تصادفی بسازید و آن را به برنامه خود دهید تا برنامه خود را تست کنید.
توجه داشته باشید که **آخرین مهلت تحویل سناریوهای اول و دوم تاریخ ۱۰ تیر و آخرین مهلت تحویل سناریو سوم قبل از امتحان پایان‌ترم به تاریخ ۳ تیر می‌باشد** .

۵ نحوه محاسبه نمرات

در این پروژه سه فاز وجود دارد که هرکدام شامل یک نمره می‌باشند. نحوه نمره‌دهی به این صورت است که در صورتی که شما کدی آپلود کنید که درست کار کند و بتواند در بازی شرکت کند، 0.25 نمره دریافت خواهید کرد. سپس، طبق جایگاهی که در فاز مربوطه در میان دانشجویان دیگر کسب می‌کنید (بر حسب دهک)، به شکل زیر نمره خواهید گرفت.

دهک	میزان نمره
10	0.75
9	0.7
8	0.65
7	0.6
6	0.55
5	0.5
4	0.4
3	0.3
2	0.2
1	0.1

به عنوان مثال اگر شما از بین 150 نفر شرکت کننده رتبه 42 را کسب کنید، دهک شما برابر هشت خواهد بود و 0.65 نمره مضاف بر 0.25 اولیه دریافت خواهید کرد.