لطفاً به نکات زیر توجه کنید:

- مهلت ارسال این تمرین ۹۸/۹/۳۰ است.
- در صورتی که به اطلاعات بیشتری نیاز دارید می‌توانید به صفحه‌ی تمرین در وب‌سایت درس مراجعه کنید.
- این تمرین شامل سوال‌های برنامه‌نویسی می‌باشد، بنابراین توجه کنید که حتماً موارد خواسته‌شده در سوال را رعایت کنید. در صورتی که به هر دلیلی سامانه‌ی داوری نتواند آن را اجرا کند مسئولیت آن تنها به عهده‌ی شماست.
- ما همواره هم‌فکری و هم‌کاری را برای حلِ تمرین‌ها به دانشجویان توصیه می‌کنیم. اما هر فرد باید تمامی سوالات را به تنهایی تمام کند و پاسخ ارسالی حتماً باید توسط خود دانشجو نوشته‌شده باشد. لطفاً اگر با کسی هم‌فکری کردید نام او را ذکر کنید. در صورتی که سامانه‌ی تطبیق، تقلبی را تشخیص دهد متأسفانه هیچ مسئولیتی بر عهده‌ی گروه تمرین نخواهد بود.
- لطفاً برای ارسال پاسخ‌های خود از راهنمای موجود در صفحه‌ی تمرین استفاده کنید.
- هر سوالی درباره‌ی این تمرین را می‌توانید در گروه درس مطرح کنید و یا از دستیاران حلِ تمرین بپرسید.

سوالات عملی

۱. درخت جست و جوی مینی مکس (۲۵ نمره)

در کلاس، الگوریتم جست و جوی مینی مکس را یاد گرفتیم. حال می خواهیم آن را پیاده سازی کنیم. در این سری از تمرین ها دوباره با فریم ورک پک من کار خواهیم داشت. لطفا فایل زیپ را از صفحه‌ی تمرین دانلود و آن را از حالت فشرده خارج کنید.
هدف این سری، آزمایش و پیاده سازی الگوریتم مینی مکس در محیط پک من است تا به عامل هایی با رفتار هوشمندانه تر برسیم.
وظیفه ی شما تنها پیاده سازی یک عامل هوشمند است که کنترل شخصیت پک من یا یکی از روحها را بر عهده می گیرد. کلاس Agentبه همین منظور تعبیه شده است. در هر مرحله موتور بازی وضعیت همه‌ی المان های بازی را محاسبه می کند و سپس با فراخوانی متد getAction از این کلاس و همچنین پاس دادن وضعیت زمین به آن، حرکت بعدی عامل را درخواست می کند. موتور بازی این روند را برای تمامی عامل های موجود (پک من و تمامی روحها) به ترتیب اعمال می کند. پس از اینکه آخرین عامل حرکت خود را انجام داد موتور بازی دوباره به عامل اول باز می گردد بدین ترتیب یک مرحله از بازی سپری می شود.
حال می‌خواهیم عاملی پیاده سازی کنیم که در هر مرحله با استفاده از الگوریتم مینی مکس امتیاز همه‌ی حرکت های مجاز را به دست بیاورد و سپس از بین آنها بهترین را انتخاب کند. در این سری از سوال ها فرض می کنیم فقط دو عامل در زمین وجود دارند: پک من و یک روح. بنابراین هر لایه از درخت مینی مکس فقط مربوط به یکی از این عامل هاست.

برای پاسخ به این سوال باید بدنه تابع minimax() را از کلاس MinimaxAgent در فایل adversarialAgent.py پر کنید. خروجی تابع، مقدار مینی مکس مربوطه است. توضیحات هر کدام از پارامترها در کد آمده است. برای آزمایش درستی کد خود از دستور زیر استفاده کنید:

```
$ python autograder.py -q q1
```

اگر کد شما درست باشد باید امتیاز ۲۵ از ۲۵ بگیرد. اما برای اینکه عملکرد این عامل را در بازی ببینیم از دستور زیر استفاده کنید. دقت کنید که مهم ترین بخش در عامل های بر پایه ی مینی مکس، تابع ارزیابی آن است. این تابع با ورودی گرفتن یک حالت از بازی، میزان مناسب بودن آن حالت را از نگاه عاملی که برای آن نوشته شده است خروجی می دهد. اما در این سوال از تابع ارزیابی پیش فرض (و البته ضعیف) استفاده می کنیم. بنابراین ممکن است در بعضی حالات حتی پک من ببازد و یا بی هدف در زمین بماند. در سوال های بعد تابع ارزیابی قوی تری پیاده سازی می کنیم.

```
$ python pacman.py -l smallClassic -k 1 -p MinimaxAgent
```

لطفا دقت کنید که کد شما حتما باید عمق درخت را از متغیر self.depth بخواند. مطمئن شوید پیاده سازی شما عمومی است و می‌توان از آن هم در پک من و هم در روح ها استفاد کرد، برای این کار می‌توانید از self.index و self.getOpponentIndex() استفاده کنید.

۲. پک‌من واقعا هوشمند (۲۵ نمره)

در سوال قبل الگوریتم مینی مکس را پیاده سازی کردیم. اما برای اینکه پک من واقعا رفتار هوشمندی از خود نشان دهد باید تابع ارزیابی قوی تری برای آن داشته باشیم. همان طور که گفته شد وظیفه ی تابع ارزیابی، بررسی میزان خوب بودن یک حالت بازی از نگاه اجرای کننده ی آن در این سوال پک من) است. به طور مثال هرچه در یک حالت، تعداد غذاهای خورده شده، میزان نزدیکی به سایر غذاها و ... بهتر باشد باید خروجی تابع ارزیابی بزرگ تر شود.

برای پاسخ به این سوال باید بدنه تابع evaluationFunction() را از کلاس SmartPacmanAgent در فایل adversarialAgents.py پر کنید. خروجی تابع، عددی برای میزان خوب بودن حالت ورودی است. توضیحات هر کدام از پارامترها در کد آمده است. برای اجرای کد خود از دستور زیر استفاده کنید:

```
$ python pacman.py -l smallClassic -k 1 -p SmartPacmanAgent -a depth=2
```

کد شما نباید در هیچ حالتی ببازد و همچنین باید تمامی غذاهای روی زمین را هم بخورد. اما ارزیابی اصلی کد شما با استفاده از روح های هوشمندتری خواهد بود. حال برای آزمایش نهایی کد خود دستور زیر را اجرا کنید:

```
$ python pacman.py -l smallClassic -k 1 -p SmartPacmanAgent -g DirectionalGhost
```

برای دریافت امتیار کامل از این سوال، کد شما باید از هر ۱۰ بار اجرای بازی حداقل ۸ بار آن را برنده شود. برای پیاده‌سازی خود می‌توانید از توابع و داده ساختار های موجود در util.py استفاده کنید.

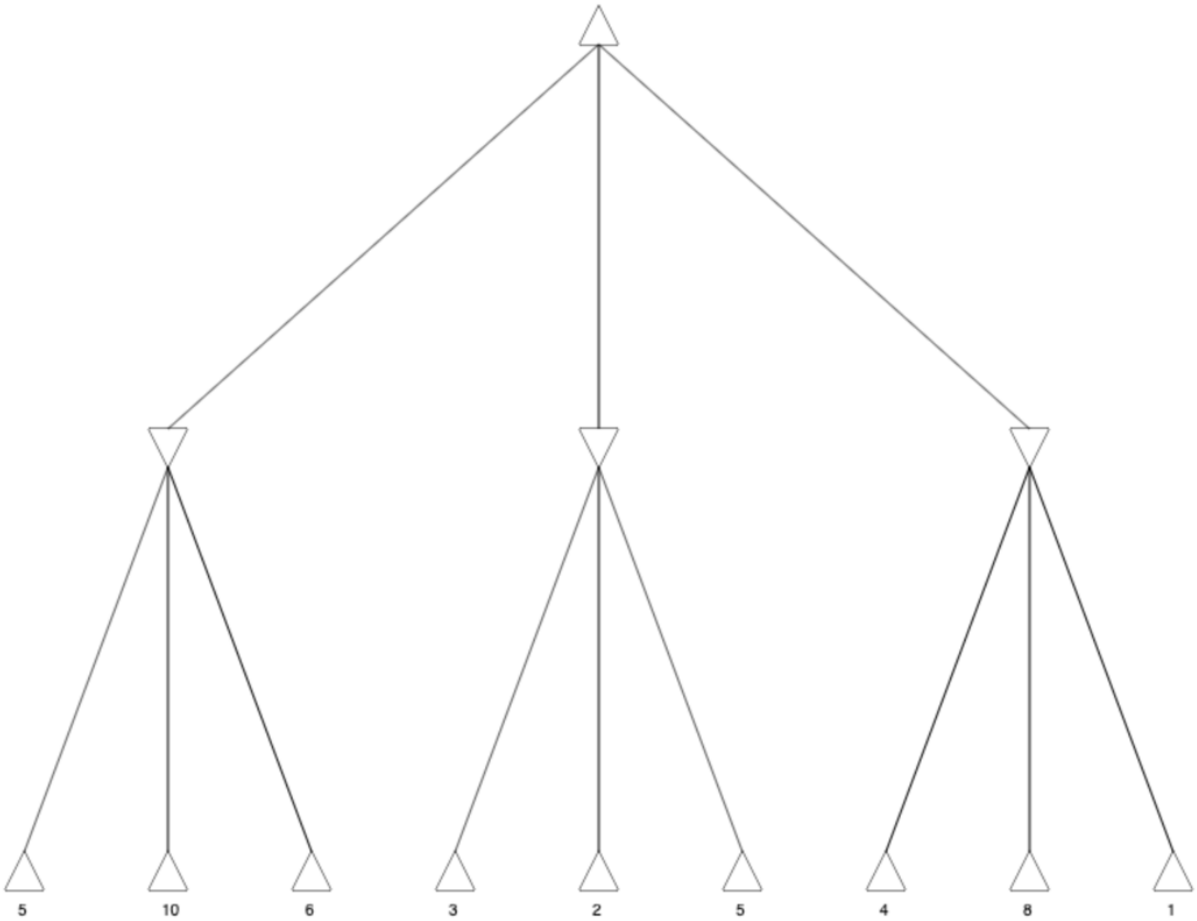
سوالات تئوری

1. الگوریتم Minimax و هرس Alpha-beta را در کلاس یاد گرفته‌اید با توجه به آن به سوالات زیر پاسخ دهید.(۱۰ نمره)

الف) پیچیدگی زمانی و حافظه الگوریتم Minimax را محاسبه کنید و توضیح دهید.

ب) پیچیدگی زمانی و حافظه الگوریتم Minimax به همراه هرس Alpha-beta را محاسبه کنید و توضیح دهید.

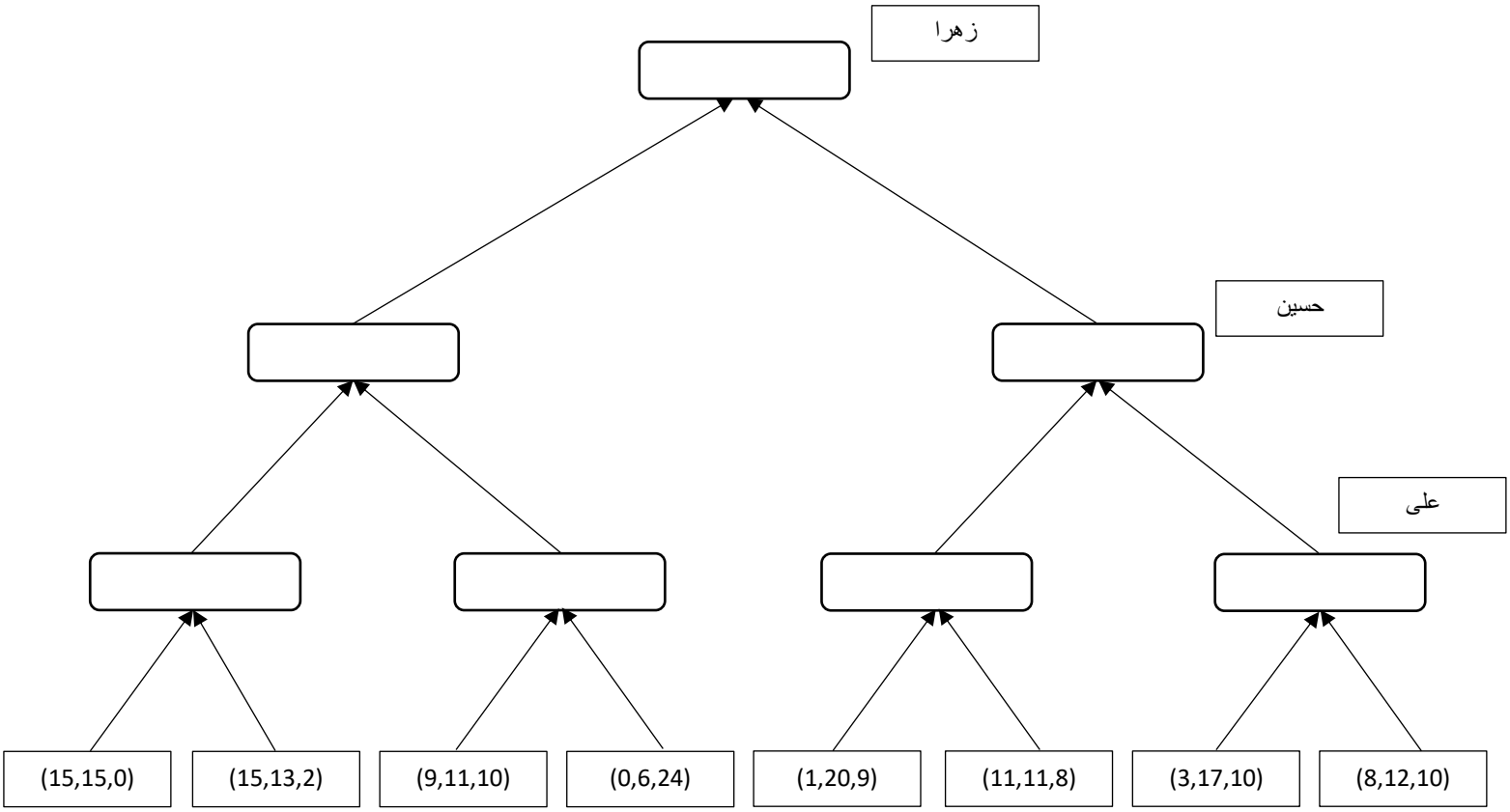
2. هرس Alpha-beta را بر روی درخت زیر اجرا کنید و در هر مرحله Alpha و Beta را مشخص کنید. (۱۰ نمره)



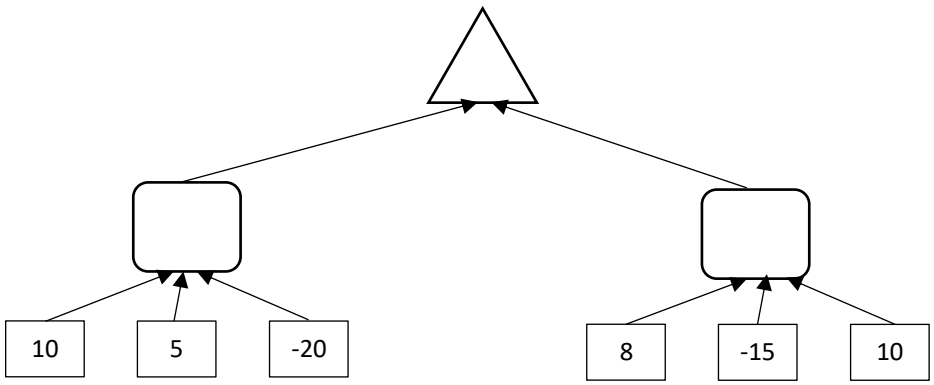
3. زهرا،حسین و علی سه دوست هستند که تعدادی شیرینی را بین خودشان میخواهند تقسیم کنند. هر کدام سهم بیشتری از شیرینی ها را میخواهند. شیرینی ها به صورت های مختلفی به سه قسمت قابل تقسیم شدن هستند که به شکل یک سه‌تایی نمایش داده میشوند. به عنوان مثال (۱۰,۱۵,۵) یک تقسیم بندی میباشد که در آن به زهرا ۱۰،به حسین ۱۵ و به علی ۵ شیرینی میرسد و در مجموع ۳۰ شیرینی وجود دارد.

در شکل زیر تقسیم بندی های مختلف شیرینی ها در برگ های یک درخت Minimax نشان داده شده است همچنین به ترتیب علی، حسین و زهرا در درخت Minimax میتوانند از بین دسته‌بندی های سطر بعدی انتخاب کنند:(۱۰ نمره)

الف) با توجه به اینکه هر سه نفر تمایل به گرفتن تعداد بیشتری شیرینی دارند مقدار شیرینی که به هر کدام از آن ها میرسد را محاسبه کنید.
ب) هرس Alpha-beta را بر روی این درخت انجام دهید و در صورتی که مقداری از درخت حذف میشود آن را مشخص کنید.



4 .دو بازیکن یک و دو با هم مسابقه میدهند.بازیکن یک همواره بهترین بازی خود را انجام میدهد اما بازیکن دو اشتباه میکند و در ۱/۳ مواقع به نفع بازیکن یک عمل میکند و در ۲/۳ موارد به نفع خودش بازی میکند.(بازیکن یک امتیاز را زیاد و بازیکن دو امتیاز را کم میکند)(۱۰ نمره)
الف) با توجه به این فرض مقادیر خانه های خالی در درخت زیر را بنویسید.



ب) درخت بدست آمده در بالا را فقط با استفاده از عناصر درخت Expectimax بازنویسی کنید.

5 .در درخت زیر بازیکن قصد دارد امتیاز خود را بیشینه کند امتیاز اینکه سمت چپ درخت را انتخاب کند ۱۵ میباشد و از امتیاز انتخاب سمت راست درخت خبر نداریم و فقط میدانیم یکی از امتیاز های ۱۵،۴۹-و۱۰- با احتمال یکسان در صورت انتخاب سمت راست درخت کسب میشود.

بازیکن به سه شکل میتواند عمل کند: میتواند سمت چپ درخت را انتخاب کند یا سمت راست درخت را انتخاب کند یا با صرف هزینه c=۱ واحد ابتدا سمت راست درخت را ببیند و امتیازی که میتواند کسب کند را محاسبه و سپس بین چپ یا راست درخت انتخاب کند. با این فرض به سوالات زیر پاسخ دهید: (۱۰ نمره)

الف) اگر بازیکن به روش آخر بازی کند ارزش خانه ۱ را بدست آورید.

ب) برای چه بازه‌ای از c انتخاب روش آخر برای بازیکن بهینه میباشد.