نیم‌سال دوم سال ۹۷-۹۸

تمرین سری سوم: مسائل جست‌وجوی رقابتی

لطفاً به نکات زیر توجه کنید:

- مهلت ارسال این تمرین تا ۱۶ اردیبهشت 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 از دستیاران حلِ تمرین بپرسید.

- آدرس گروه درس: https://groups.google.com/forum/#!forum/ai972
- صفحه تمرین: https://quera.ir/course/assignments/9134/problems

موفق باشید

سوال‌های عملی

در این تمرین به پیاده‌سازی عامل هوشمند برای بازی Othello پرداخته میشود.
برای کسب اطلاعات بیشتر در رابطه با این بازی میتوانید به لینک زیر مراجعه کنید.
https://en.wikipedia.org/wiki/Reversi

در این پیاده سازی بازی هنگامی که هر کدام از بازیکنان حرکت مجازی برای انجام نداشته باشند خاتمه می یابد.
هر state در این بازی شامل بازیکن در حال بازی و همچنین اطلاعات مربوط به مهره‌های موجود در صفحه میباشد. مهره های سیاه با عدد 1 و مهره های سفید با عدد 2 و خانه های خالی با عدد 0 نمایش داده میشوند.
به عنوان مثال state شروع در بازی 8*8 به شکل زیر میباشد:

```
((0, 0, 0, 0, 0, 0, 0, 0),
 (0, 0, 0, 0, 0, 0, 0, 0),
 (0, 0, 0, 0, 0, 0, 0, 0),
 (0, 0, 0, 2, 1, 0, 0, 0),
 (0, 0, 0, 1, 2, 0, 0, 0),
 (0, 0, 0, 0, 0, 0, 0, 0),
 (0, 0, 0, 0, 0, 0, 0, 0),
 (0, 0, 0, 0, 0, 0, 0, 0))
```

برای اجرای بازی می توان از دستور زیر استفاده کرد که دو آرگومان به عنوان عامل هوشمند میگیرد.

```
Python3 othello_gui.py random_ai.py random_ai.py
```

همچنین در صورت ندادن آرگومان کاربر را به عنوان بازیکن در نظر میگیرد.

تذکر:

- **شما فقط مجاز به تغییر فایل ai.py می‌باشید.**
- **همچنین در صورتی که برنامه شما در کمتر از ۱۰ ثانیه دستوری ارسال نکند بازی به نفع عامل دیگر پایان می‌یابد.**

سوالات عملی

۱- الگوریتم مینی‌مکس (۲۵ نمره)

در کلاس الگوریتم مینی‌مکس را یاد گرفته‌اید. در این سوال هدف پیاده سازی الگوریتم مینی‌مکس برای بازی Othello می باشد. در ابتدا برای کوتاه شدن زمان اجرا بازی dimension را در خط 137 از فایل othello_gui.py 4 قرار بدهید.

تابع compute_utility(board, color) را در فایل ai.py به گونه ای تغییر بدهید که تفاضل تعداد مهره‌ها با رنگ مخالف color را از تعداد مهره‌ها با رنگ color برگرداند.

(خروجی تابع get_score(board) یک tuple میباشد که خانه اول آن تعداد مهره‌های سیاه و خانه دوم آن تعداد مهره‌های سفید میباشد.)

سپس در سه تابع select_move_minimax, minimax_min_node, minimax_max_node در فایل ai.py الگوریتم مینی‌مکس را پیاده سازی کنید.

همچنین میتوانید از توابع get_possible_moves و play_move در فایل othello_shared.py استفاده کنید.

برای اجرا کد از دستور زیر استفاده کنید:

```
Python3 othello_gui.py ai.py ai.py
```

در صورت بهینه بازی کردن هر دو طرف(minimax)همواره باید نفر دوم شروع کننده برنده شود.

کد شما برای دریافت نمره باید در صفحه 4*4 همواره بازیکن دوم برنده باشد.

۲- هرس $\alpha\beta$ (۲۵ نمره)

کد قسمت قبل در صفحه با ابعاد بیشتر از 4 به دلیل بزرگ شدن درخت مینی‌مکس کاربردی نیست. برای بهبود این الگوریتم از روش هرس alpha-beta استفاده میکنیم.

در توابع select_move_alphabeta, alphabeta_min_node, alphabeta_max_node الگوریتم مینی‌مکس با هرس alpha-beta را پیاده سازی کنید.

برای اجرا کد خود select_move_minimax در انتهای تابع run_ai را به select_move_alphabeta تغییر دهید.

کد شما برای دریافت نمره باید در صفحات بزرگتر عملکرد بهتری نسبت به مینی‌مکس سوال قبل داشته باشد.

۳- نگه داشتن state (۱۰ نمره)

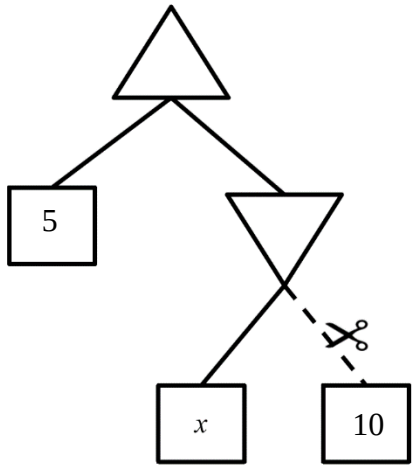
یک دیکشنری در فایل ai.py درست کنید و توابع دو سوال قبل را طوری تغییر دهید که مقدار مینی‌مکس state های مختلف را در این دیکشنری نگه دارد و در صورت وجود یک state در دیکشنری دوباره مقدار آن state را محاسبه نکند.

سوال‌های تئوری

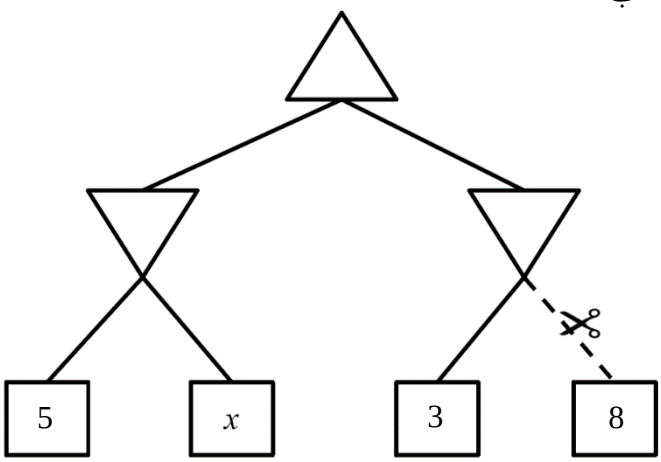
سوال اول (۱۵ نمره)

برای هر یک از درخت‌های زیر که نشان‌دهنده‌ی وضعیتی از یک بازی می‌باشند، مقادیر x را به گونه‌ای تعیین کنید که شاخه‌ی تعیین شده توسط الگوریتم $\alpha\beta\ pruning$ بررسی نشود. راه حل خود را برای پیدا کردن مقادیر x توضیح دهید.

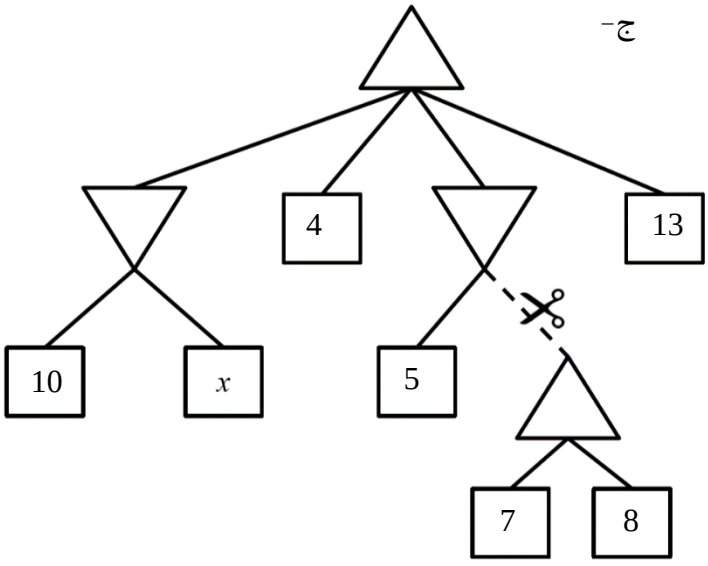
الف-



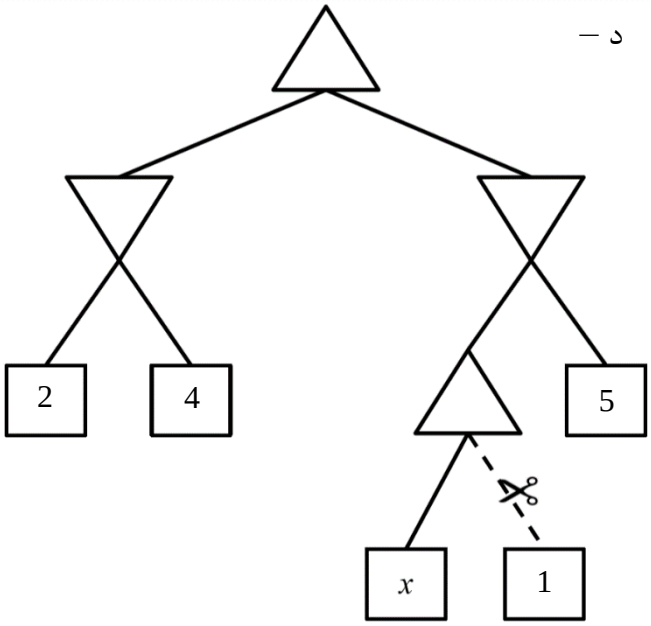
ب-



ج-

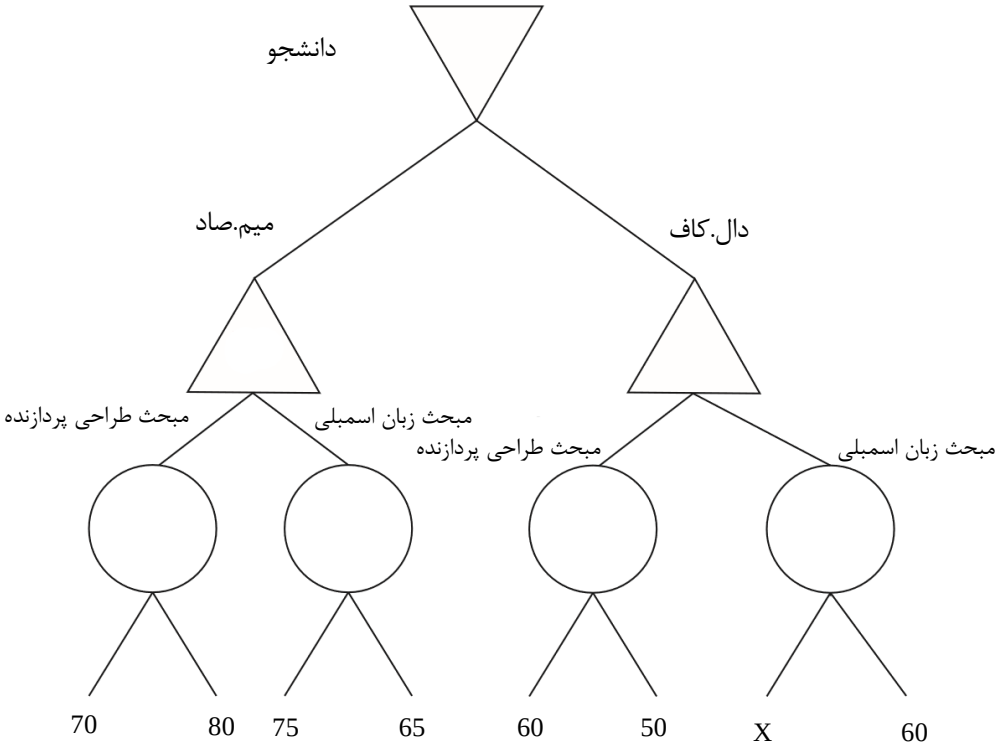


د –



سوال دوم (۱۵ نمره)

شما به عنوان یک با تجربه در ارائه پروژه‌های درسی به دست‌یاران حل تمرین، قصد کمک به دانشجویان سال پایینی خود برای درس مدار منطقی را دارید. برای پروژه‌ی این درس دانشجویان می‌توانند پروژه‌ی خود را به یکی از دو نفر میم.صاد یا دال.کاف ارائه دهند. پروژه شامل دو بخش است. اگر بدانیم برای هر بخش ۲ سوال وجود دارد که توسط TA ها از دانشجویان پرسیده می‌شود، با توجه به برآورد دانشجو از نحوه عملکرد خود در هر بخش و نحوه‌ی ارائه گرفتن TA مورد نظر، به سوالات زیر پاسخ دهید:



الف- مقادیری که به X وابسته نیستند را تکمیل کنید.

ب- با چه مقداری از X دانشجو دال.کاف را برای ارائه‌ی پروژه‌ی خود انتخاب می‌کند؟

ج- در صورتی که مقدار X برابر با ۶۰ باشد، دانشجو باید کدام TA را برای ارائه‌ی پروژه‌ی خود انتخاب کند؟

د- در حالت قبلی دانشجو TA را برای ارائه‌ی پروژه‌ی خود انتخاب می‌کند. برای حالتی که دانشجو حق انتخاب مبحث را داشته باشد و TA ارائه گیرنده به صورت تصادفی انتخاب شود، درخت Expectimax را رسم کنید.