بسمه تعالی

مستند مسابقه - نسخه نهایی

یازدهمین نبرد هوش مصنوعی شریف

مرحله‌ی حضوری

اسفند ۱۳۹۷

آخرین نبرد

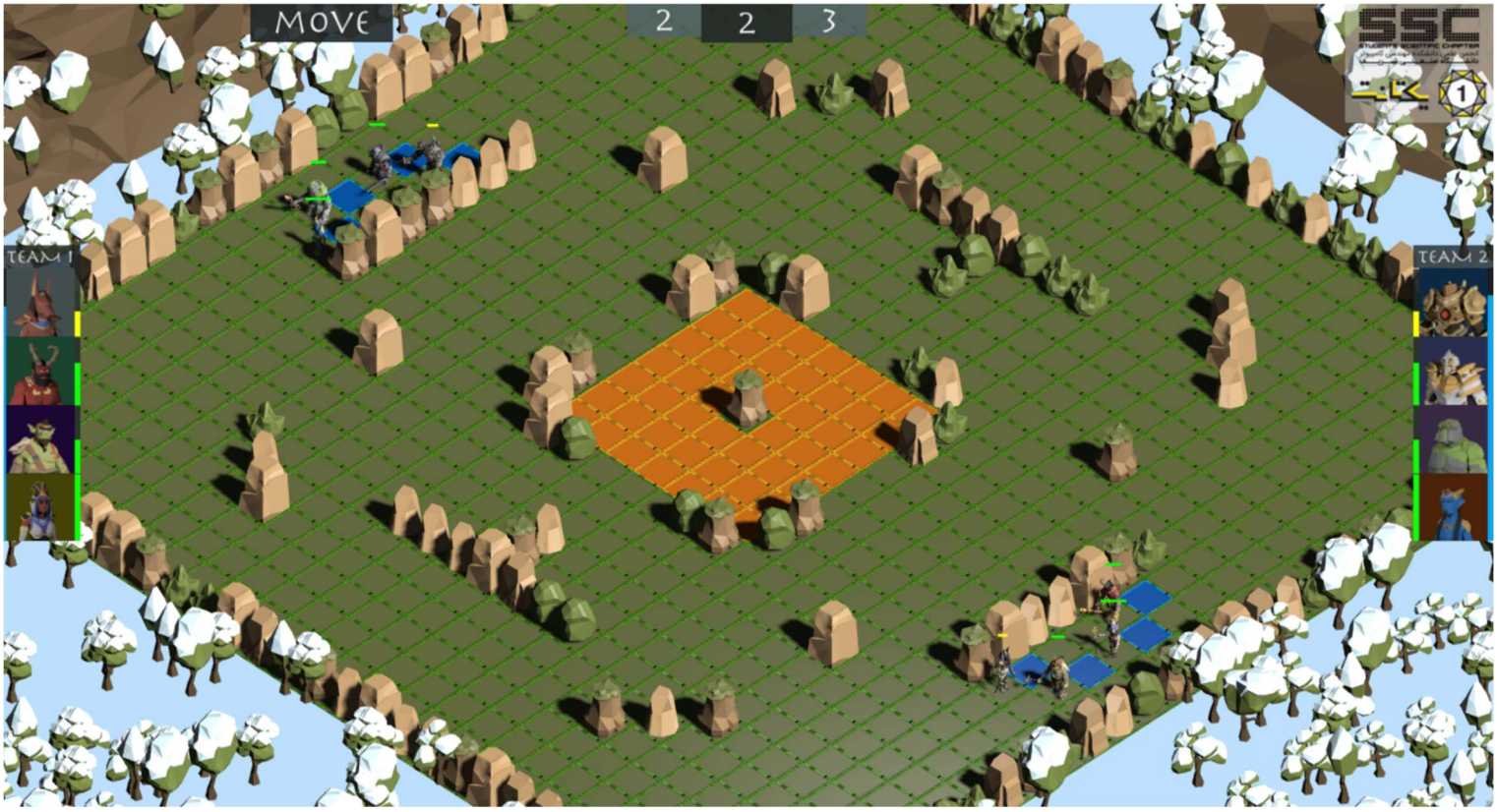
در زمان‌های بسیار قدیم، آشوب نیز همانند آب، خاک و هوا عنصری از عناصر طبیعت بود. در آن زمان موجوداتی ساکن زمین بودند که با استفاده از قدرت این آشوب می‌توانستند کارهای خارق‌العاده‌ای انجام دهند؛ برخی از این موجودات خیر بودند و از سایر موجودات زنده محافظت می‌کردند، برخی دیگر نیز طرف شر را گرفته و هدفی جز تخریب و نابودی سایر موجودات نداشتند.

امروزه در افسانه‌ها از این آشوب، با کلماتی مانند جادو و سحر نام برده می‌شود. همچنین گروه خیر در افسانه‌ها به نام‌های قهرمانان و اسطوره‌ها ثبت شدند و گروه شر نام‌هایی چون دیوها و شیاطین را گرفتند. مادامی که جادو در زمین فراوان بود، مبارزه بین این دو گروه خیر و شر همواره ادامه داشت و هیچ یک از دو گروه نمی‌توانست گروه دیگر را به طور کامل شکست دهد و کنترل زمین را به عهده بگیرد.

اما با گذشت زمان، شرایط زمین رو به تعادل می‌رفت و جادو که ماهیت آن همان آشوب بود، به مرور در زمین کمرنگ شد و با سایر عناصر طبیعت ترکیب شد. با تضعیف جادو در زمین، موجودات جادویی نیز کم کم قدرت خود را از دست دادند و یکی یکی در مکان‌های مختلف به خواب فرو رفتند. زندگی روی زمین بدون جادو و موجودات جادویی ادامه داشت و رفته‌رفته انسان‌ها پیشرفت کرده و با رشد تکنولوژی و علم، زمین را در اختیار خود درآوردند. اما این رشد بدون توجه به طبیعت باعث شد که انسان‌ها صدمات بسیاری به طبیعت وارد آورند و تعادل موجود در آن را به هم بزنند. فشارهای انسان‌ها به طبیعت ادامه داشت تا

زمانی که این تعادل به هم خورد و آشوب یا جادو که در تمام این مدت به بخشی از طبیعت تبدیل شده بود، همانند آتش‌فشانی در نقطه‌ای از زمین فوران کرد.

با ظهور دوباره‌ی جادو در زمین، موجودات افسانه‌ای که به خواب رفته بودند، بار دیگر برخاستند تا این نیروی عظیم را تصاحب کنند. در تمام این مدت، جادو که به بخشی از طبیعت تبدیل شده بود، به شکلی دست‌نخورده باقی مانده بود؛ و به همین علت حجم نیروی فورانی به حدی بود که هم خیر و هم شر می‌توانستند با استفاده از آن نیروی مقابل را به طور کامل نابود کنند و به این جنگ طولانی پایان ببخشند. قهرمانان هر دو طرف به سرعت به مکان فوران جادو رفتند تا در آن‌جا مبارزه‌ای نهایی انجام دهند، بر سر نیرویی که می‌توانست این جنگ طولانی را به اتمام برساند.



وضعیت بازی:

فاصله: پیش از هر چیز لازم است مفهوم فاصله در این بازی بیان شود. هرگونه فاصله در این بازی، جمع اختلاف شماره‌ی سطرها و اختلاف شماره‌ی ستون‌های دو خانه (برای قهرمانان خانه‌ی فعلی آن‌ها) می‌باشد. به این نوع محاسبه‌ی فاصله، فاصله‌ی **منهتنی** گفته می‌شود.

تیم‌ها: در هر مسابقه، تیم‌های شرکت کننده در قالب دو تیم با یک‌دیگر به رقابت می‌پردازند. هر تیم شامل ۴ قهرمان می‌باشد. ویژگی‌های تیم به شرح زیر است:

- **امتیاز عمل (AP):** هر تیم مشخصه‌ای تحت عنوان "امتیاز عمل" دارد که برای استفاده از قابلیت‌ها و حرکت قهرمانان خود، از این مشخصه هزینه می‌کند. در انتهای هر نوبت، مقدار این مشخصه به حداکثر خود بازمی‌گردد.
- **دید (Vision):** تمامی خانه‌های معمولی نقشه که در دید قهرمانان تیم قرار دارد، برای کل تیم قابل مشاهده است.

نوبت (Turn): بازی به صورت نوبتی انجام می‌شود. هر نوبت متشکل از ۷ فاز می‌باشد. ۶ فاز اول فاز حرکت (Move Turn) و فاز آخر هر نوبت فاز عمل (Action Turn) می‌باشد. در ادامه توضیحات کامل‌تری نسبت به فازها ارائه خواهد شد.

دو تیم روی یک نقشه به مبارزه می‌پردازند. نقشه‌ی بازی به شکل یک آرایه‌ی دو بعدی از خانه هاست. اولین درایه را سطر و دومین درایه را ستون می‌نامیم. خانه‌ی مبدا (۰،۰) نیز در بالا و چپ نقشه قرار دارد.

در هر لحظه اجزای زیر در نقشه‌ی بازی وجود دارد:

خانه های نقشه:

خانه های معمولی (Normal Cells): قهرمانان دو تیم می‌توانند به این خانه‌ها وارد شوند. با این محدودیت که در هر لحظه، حداکثر یک قهرمان از هر تیم در یک خانه‌ی مشخّص می‌تواند وجود داشته باشد. در یک خانه دو قهرمان از تیم‌های متفاوت به طور هم‌زمان می‌توانند حضور داشته باشند. مرزهای خانه‌ی معمولی جزء آن ***محسوب نمی‌شود***.

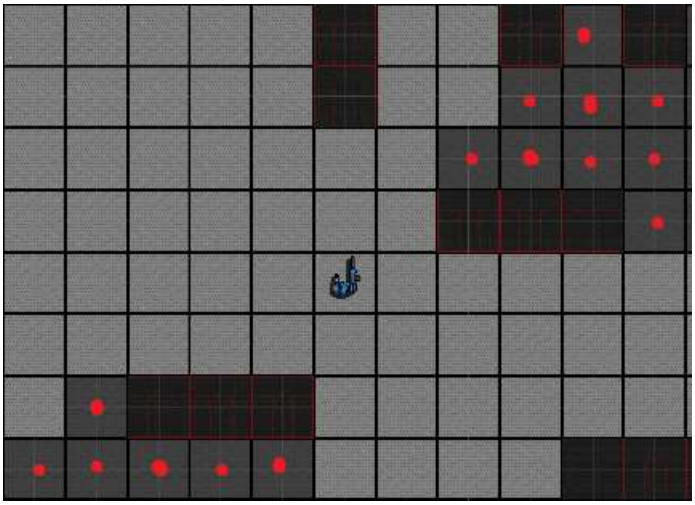
خانه های مسدود (Wall Cells): هیچ قهرمانی به این خانه‌ها نمی‌تواند وارد شود. هم‌چنین این خانه‌ها، دید قهرمانان و مسیر قابلیت‌های خطّی (که در قسمت قابلیت‌ها توضیح داده خواهد شد) را مسدود می‌سازند. مرزهای خانه‌ی مسدود جزء آن ***محسوب می‌شود***.

منطقه های خاص:

منطقه‌ی هدف (Objective Zone): مجموعه‌ای از خانه‌های نقشه که حضور قهرمان در هر نوبت در آن، برای تیم امتیاز به همراه دارد.

منطقه‌ی زنده شدن (Respawn Zone): به ازای هر تیم یکی از این محدوده‌ها در نقشه وجود دارد. قهرمان‌ها پس از مرگ و گذشتن مدت مرگ، به تصادف در یکی از خانه‌های این محدوده زنده می‌شوند. این منطقه، منطقه‌ای امن است و نمی‌توان از داخل آن به خارج و برعکس آسیبی زد.

قهرمانان (Heroes): قهرمانان در ۵ کلاس نگهبان (Guardian)، درمانگر (Healer)، ویرانگر (Blaster)، قراول (Sentry) و سایه (Shadow) می‌باشند. هرکلاس از قهرمان‌ها، قابلیت‌های ویژه‌ی خود را دارد.



ویژگی‌های هر قهرمان به شرح زیر است:

دید (Vision):

یک قهرمان به یک خانه دید دارد اگر در میان خط واصل مرکز خانه‌ی کنونی و مرکز خانه‌ی مقصد، هیچ خانه‌ی مسدودی نباشد. در تصویر مقابل، خانه‌هایی که با دایره‌ی قرمز مشخص شده‌اند، خارج از محدوده‌ی دید قهرمان آبی رنگند. خانه‌های مشکی نیز مسدودند.

سلامتی اولیه (Max HP):

مقدار اولیه‌ی سلامتی قهرمان پس از زنده شدن.

سلامتی فعلی (Current HP):

میزان سلامت فعلی قهرمان. با رسیدن به صفر یا کمتر قهرمان می‌میرد.

مدت زمان زنده شدن (Respawn Time):

پس از مرگ قهرمان، با گذشتن این تعداد نوبت، قهرمان در خانه‌ای از منطقه‌ی زنده شدن تیم خودی به تصادف زنده می‌شود.

مدت زمان زنده شدن باقی‌مانده (Remaining Respawn Time) :

در صورت مرده بودن، قهرمان پس از این تعداد نوبت زنده خواهد شد.

هزینه‌ی حرکت (Move Cost):

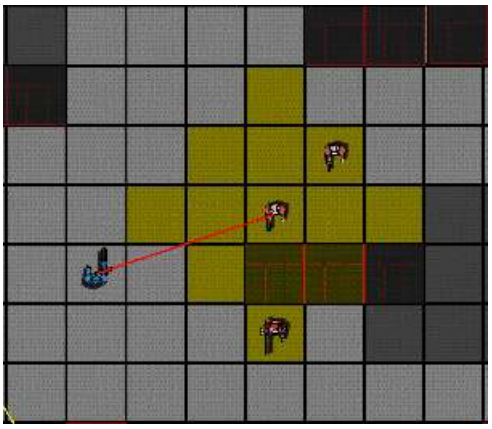
به ازای هر خانه حرکت قهرمان، به این میزان از امتیاز عمل بازیکن کم می‌شود.

قابلیت (Ability):

هر قهرمان ۳ قابلیت دارد. مشخصات هر قابلیت به صورت زیر است:

- **نوع قابلیت (Ability Type):** نوع قابلیت به یکی از سه شکل زیر است و به همین ترتیب نیز اعمال می‌شوند:
  - **دفاعی (Defensive):** قابلیت هایی که بر قهرمانان خودی اثر می‌گذارد. مانند افزایش سلامتی.
  - **گریز (Dodge):** جابجایی سریع و آنی خود قهرمان در نقشه. با مشخص کردن یک خانه، قهرمان در همین نوبت به آن خانه جابجا می‌شود. از این قابلیت‌ها برای گریز از حمله‌ی دشمن می‌توان استفاده کرد.
  - **تهاجمی (Offensive):** مهارت هایی که بر قهرمانان حریف اثر می‌گذارد. مانند کاهش سلامتی.
- **هزینه‌ی انجام (Ability Cost):** با انجام قابلیت، این میزان از امتیاز عمل بازیکن کم می‌شود.
- **مدت زمان غیرفعال (Cooldown):** پس از استفاده، قابلیت پس از این تعداد نوبت قابل استفاده می‌شود.

- **قدرت (Power):** مقدار اثرگذاری قابلیت. برای قابلیت‌های تهاجمی میزان آسیب وارده به حریف و برای قابلیت‌های تدافعی میزان افزایش سلامتی قهرمان خودی می‌باشد.



- **برد اثرگذاری (Area Of Effect):** تا این فاصله از خانه‌ی هدف را تحت تاثیر قرار می‌دهد. برای مثال قابلیتی با برد اثرگذاری ۲، بر تمامی خانه‌هایی که مجموع فاصله‌ی سطر آن از سطر خانه‌ی هدف و فاصله‌ی ستون آن از ستون خانه‌ی هدف کوچکتر مساوی ۲ باشد، اعمال می‌شود. به تصویر این مثال توجه کنید. خانه های زرد رنگ تحت تاثیر قرار می‌گیرند.:
- **حداکثر فاصله اعمال (Range):** حداکثر فاصله از قهرمان که می‌توان به عنوان خانه‌ی هدف قابلیت انتخاب کرد.
- **پرتابی/خطّی (Is Lobbing) :** قابلیت‌های پرتابی مستقیماً در خانه‌ی هدف اعمال می‌شوند و در واقع چیزی در میان مسیر نمی‌تواند جلوی آن‌را بگیرد. برای قابلیت‌های خطی، خط واصل مرکز خانه‌ی مبدا و مرکز خانه‌ی هدف را در نظر بگیرید. این خط تا جایی ادامه پیدا می‌کند که به یک قهرمان یا خانه‌ی مسدود برسد. در صورت رسیدن به قهرمان، قابلیت در همان خانه‌ی قهرمان اعمال می‌شود. در صورت رسیدن به خانه‌ی مسدود، قابلیت در خانه‌ی معمولی پیش از آن اعمال می‌شود. در غیر این دو صورت، قابلیت در خانه‌ی هدف تعیین شده‌ی خود اعمال می‌شود.
- **شکافندگی (Is Piercing):** این ویژگی مختص قابلیت‌های خطی است. قابلیت خطی شکافنده، توسط قهرمانان هدف خود متوقف نمی‌شود و پس از اثرگذاری بر هر قهرمان در مسیر خود، به مسیر خود تا انتها یا برخورد با دیوار ادامه می‌دهد و در تمام طول مسیر به اثرگذاری ادامه می‌دهد.

قابلیت اعمال شده (Cast Ability): هر تیم به قابلیت‌های اعمال شده در نوبت قبلی بازی دسترسی دارد. اما این دسترسی همیشه به طور کامل وجود ندارد. در صورت تحقق حداقل یکی از شرط های زیر دسترسی به نام قابلیت (Ability Name) نیز وجود خواهد داشت:

- در صورت حضور قهرمان اعمال کننده در دید ما، شناسه‌ی این قهرمان (Caster ID) معتبر است.
- در صورت حضور قهرمان اعمال کننده در ابتدای اعمال قابلیت در دید ما، خانه‌ی آغاز (Start Cell) معتبر است.
- در صورت مشاهده‌ی خانه‌ی هدف در دید ما، خانه‌ی هدف (End Cell) معتبر است.
- به شناسه‌ی تمامی قهرمانانی که این قابلیت بر آن‌ها اثر داشته و در دید ما حضور داشته‌اند، دسترسی داریم (Target Hero Ids).

برای درک بهتر به دو مثال زیر توجه کنید:

- فرض کنید در نوبت پیشین، قهرمان ویرانگر ما یک بمب به پشت خانه‌ی مسدودی انداخته باشد که آن سمت در محدوده‌ی دید ما نیست. حال حتی اگر در آن محدوده قهرمانی از حریف باشد، چون در دید ما نبوده است، با getTargetHeroIds شناسه‌ی هیچ قهرمانی مشاهده نمی‌شود.

- در همین مثال قبلی، اگر قهرمان ویرانگر ما نیز در محدوده‌ی دید حریف نباشد، حریف چون خانه‌ی هدف بمب را دیده است (قهرمانش در آن خانه حضور داشته)، در getCastAbilities این قابلیت بمب را مشاهده می‌کند. اما چون قهرمان ویرانگر ما در محدوده‌ی دید او نیست، Start Cell و Caster ID برای او نامعتبر است.

روند بازی:

پیش‌پردازش (PreProcess): پیش از شروع بازی، طی یک نوبت به هرتیم اجازه‌ی انجام پیش‌پردازش‌هایی داده می‌شود.

فاز انتخاب (Pick Turn): در ابتدای بازی، دو طرف به انتخاب قهرمان‌های خود می‌پردازند. این انتخاب در ۴ مرحله به صورت هم‌زمان صورت می‌گیرد و در پایان هر مرحله، قهرمانان انتخاب شده تا اینجا به دو طرف معرّفی می‌شوند. دقت شود که انتخاب قهرمانان یکسان **مجاز است.**

پس از پایان فاز انتخاب، بازی آغاز می‌شود. هر نوبت بازی شامل ۶ فاز حرکت و ۱ فاز عمل است. پس از اتمام هر نوبت، امتیاز عمل هر تیم به مقدار اولیه باز می‌گردد.

فازهای حرکت (Move Turn): در هرکدام از این فازها، هر قهرمان را می‌توان حداکثر به یکی از خانه های مجاور ضلعی جابه‌جا کرد. پس از تصمیم گیری دو طرف و در انتهای هر فاز، حرکات به شکل هم‌زمان صورت می‌گیرند.

فاز عمل (Action Turn): در این فاز، حداکثر یک قابلیت از هر قهرمان را می‌توان استفاده کرد. در انتهای فاز، به ترتیب نوع قابلیت‌ها، یعنی ابتدا دفاعی‌ها، سپس گریزها و در انتها تهاجمی‌ها اعمال می‌شوند.

در انتهای هر نوبت، به ازای تعداد قهرمانان زنده‌ی موجود در منطقه‌ی هدف، و ۱۰ برابر تعداد قهرمانان کشته شده از حریف در این نوبت، به امتیاز تیم مربوطه اضافه می‌شود.

پایان بازی:

در هر نوبت از بازی، تیم برتر تیمی است که امتیاز بیشتری دارد. امتیاز هر تیم شامل مجموع امتیاز کشتن قهرمانان حریف و مجموع امتیاز حضور در محدوده‌ی هدف می‌باشد. در صورت تساوی امتیازات، تیمی که در مجموع نوبت‌ها امتیاز عمل کمتری استفاده کرده، تیم برتر است.

تا هنگامی که یک تیم به امتیاز ۳۰۰ یا بالاتر برسد یا نوبت ۱۰۰ فرا رسد بازی ادامه می‌یابد. پس از رسیدن به هرکدام از دو مورد پیشین، اگر اختلاف امتیاز تیم برترو تیم دیگر بیشتر یا مساوی ۳۰ امتیاز باشد، بازی تمام می‌شود و تیم برتر برنده است. در غیر این‌صورت، وقت اضافه‌ای به میزان k نوبت در نظر گرفته می‌شود. در صورت حفظ برتری تا پایان این k نوبت، تیم برتر برنده اعلام می‌شود. در غیر این‌صورت، k-1 نوبت دیگر با شرایط قبلی وقت اضافه داده می‌شود تا هنگامی که یا تیمی پیروز شود یا وقت اضافه داده شده به صفر برسد. در صورت تساوی، تیمی که کد خود را زودتر ارسال کرده، برنده اعلام می‌شود.

ویژگی‌های بازی:

نام ویژگی	مقدار ویژگی
امتیاز عمل اولیه (AP)	۱۰۰
حداکثر امتیاز پیروزی (Max Score)	۳۰۰
حداکثر اختلاف امتیاز پیروزی (Max Score Diff)	۳۰
حداکثر نوبت بازی (Max Turn)	۱۰۰
مقدار اولیه نوبت اضافه (Init Overtime)	۵
تعداد فازهای انتخاب (Pick Turn)	۴
تعداد فازهای حرکت در هر نوبت (Move Turn)	۶
تعداد فازهای عمل در هر نوبت (Action Turn)	۱
امتیاز کشتن یک قهرمان از حریف (Kill Score)	۱۰
امتیاز منطقه‌ی هدف (Objective Zone Score)	۱

در ادامه به معرفی کلاس‌های مختلف قهرمان‌ها می‌پردازیم. دو تصویر هر کلاس، ظاهر قهرمانان آن کلاس در دو تیم مختلف را نشان می‌دهد و این تفاوت صرفا ظاهری می‌باشد.

ویرانگر – Blaster

Red Demon *Slayer*

توضیحات: ویرانگر نسبت به دیگر قهرمان ها از سرعت خوبی برخوردار است. قابلیت خاص این قهرمان، پرتاب بمب با قدرت آسیب و محدوده اثرگذاری بالاست که می‌توان چند قهرمان کنار هم را در یک نوبت هدف قرار داد.

ویژگی های کلی:

نام ویژگی	مقدار ویژگی
سلامتی اولیه	۲۵۰
مدت زمان زنده شدن	۵
هزینه‌ی حرکت	۴

ویژگی های قابلیت‌ها:

نام ویژگی	حمله	گریز	بمب
نوع قابلیت	تهاجمی	گریز	تهاجمی
هزینه‌ی انجام	۱۵	۲۵	۲۵
مدت زمان غیرفعال	۱	۶	۴
قدرت	۲۰	-	۳۵
برد اثرگذاری	۱	-	۲
حداکثر فاصله‌ی اعمال	۳	۴	۵
پرتابی/خطّی	خطّی	پرتابی	پرتابی
شکافندگی	خیر	-	-

قراول – Sentry

Ancient Warrior *Mechanical Golem*

توضیحات: قراول قهرمانی دوربرد اما نسبتا کند است. قابلیت خاص این قهرمان، شلیک یک اشعه با برد نامحدود و قدرت آسیب بالاست. این اشعه به اولین قهرمان یا دیوار در مسیر خود اصابت کرده و اثر خود را اعمال می‌کند.

ویژگی های کلی:

نام ویژگی	مقدار ویژگی
سلامتی اولیه	۱۲۰
مدت زمان زنده شدن	۵
هزینه‌ی حرکت	۶

ویژگی های قابلیت‌ها:

نام ویژگی	حمله	گریز	اشعه
نوع قابلیت	تهاجمی	گریز	تهاجمی
هزینه‌ی انجام	۱۵	۲۵	۲۵
مدت زمان غیرفعال	۱	۷	۵
قدرت	۳۰	-	۵۰
برد اثرگذاری	۰	-	۰
حداکثر فاصله‌ی اعمال	۷	۳	نامحدود
پرتابی/خطّی	خطّی	پرتابی	خطّی
شکافندگی	خیر	-	خیر

درمانگر – Healer

Ancient Queen

Mystic

توضیحات: درمانگر قهرمانی با سرعت و سلامتی متوسط است. قابلیت خاص این قهرمان، درمانگری است که می‌تواند سلامتی یک قهرمان خودی را حداکثر ۳۰ واحد تا مقدار سلامتی اولیه افزایش دهد.

ویژگی های کلی:

نام ویژگی	مقدار ویژگی
سلامتی اولیه	۲۰۰
مدت زمان زنده شدن	۵
هزینه‌ی حرکت	۴

ویژگی های قابلیت‌ها:

نام ویژگی	حمله	گریز	درمانگری
نوع قابلیت	تهاجمی	گریز	تدافعی
هزینه‌ی انجام	۱۵	۲۵	۱۵
مدت زمان غیرفعال	۱	۶	۲
قدرت	۲۵	-	۳۰
برد اثرگذاری	۰	-	۰
حداکثر فاصله‌ی اعمال	۴	۴	۴
پرتابی/خطّی	پرتابی	پرتابی	پرتابی
شکافندگی	-	-	-

Guardian – نگهبان

Big Ork **Elemental Golem**

توضیحات: نگهبان قهرمانی با سلامتی اولیه زیاد اما سرعتی بسیار کند است. قابلیت خاص این قهرمان، تقویت یک قهرمان خودی است که باعث می‌شود در این نوبت به مقدار نامحدود آسیب های وارده به قهرمان هدف بی اثر شود.

ویژگی های کلی:

نام ویژگی	مقدار ویژگی
سلامتی اولیه	۴۰۰
مدت زمان زنده شدن	۵
هزینه‌ی حرکت	۸

ویژگی های قابلیت‌ها:

نام ویژگی	حمله	گریز	تقویت
نوع قابلیت	تهاجمی	گریز	تدافعی
هزینه‌ی انجام	۱۵	۲۵	۲۵
مدت زمان غیرفعال	۱	۸	۷
قدرت	۴۰	-	نامحدود
برد اثرگذاری	۱	-	۰
حداکثر فاصله‌ی اعمال	۱	۲	۴
پرتابی/خطّی	پرتابی	پرتابی	پرتابی
شکافندگی	-	-	-

سایه – Shadow

Evil God

Spirit Demon

توضیحات: قابلیت خاص این قهرمان، برش است که به صورت خطی از مبدا تا مقصد خود به تمامی قهرمان های دشمن آسیب رسانده و خود نیز جابجا می‌شود. این قابلیت در دسته گریز قرار می‌گیرد. در میان مسیر برش، سایه آسیب ناپذیر است.

ویژگی های کلی:

نام ویژگی	مقدار ویژگی
سلامتی اولیه	۹۰
مدت زمان زنده شدن	۵
هزینه‌ی حرکت	۳

ویژگی های قابلیت‌ها:

نام ویژگی	حمله	گریز	برش
نوع قابلیت	تهاجمی	گریز	گریز
هزینه‌ی انجام	۱۵	۲۵	۲۰
مدت زمان غیرفعال	۱	۵	۲
قدرت	۴۰	-	۲۵
برد اثرگذاری	۰	-	۰
حداکثر فاصله‌ی اعمال	۱	۴	۶
پرتابی/خطّی	پرتابی	پرتابی	خطّی
شکافندگی	-	-	بله

جزئیات فنّی اتفاقات بازی

- در صورت نامعتبر بودن یک دستور، این دستور نادیده گرفته می‌شود و اولین دستور معتبر بعدی انجام می‌شود. ترتیب دستورات به همان ترتیبی است که شما در کلاینت ارسال می‌کنید.
- در صورتی که یک حرکت یا قابلیت به هر دلیل انجام نشود، امتیاز عمل آن از تیم کسر نمی‌شود.
- دستور حرکت با مقصد خانه‌ی مسدود (Wall Cell) یا خانه‌ی پرشده با قهرمان خودی، نامعتبر است.
- دستور قابلیت گریز با مقصد خانه‌ی مسدود (Wall Cell) یا خانه‌ی پرشده با قهرمان خودی (شامل خود قهرمان)، نامعتبر است.
- در صورت فاصله‌ی بیشتر خانه‌ی هدف قابلیت از حداکثر فاصله‌ی اعمال آن، آخرین خانه‌ی موجود در خط واصل خانه‌ی قهرمان و خانه‌ی هدف اولیه، به عنوان خانه‌ی هدف در نظر گرفته می‌شود.
- پس از اجرای اولین دستور معتبر، تمامی دستورات با قهرمان یکسان اعمال کننده با این دستور که پس از آن آمده، نامعتبر تلقی می‌شود.
- در رابطه با قابلیت‌های غیر از گریز، مقصد می‌تواند هرنوع خانه‌ای باشد و معتبر است.
- در صورت کمبود امتیاز عمل برای انجام یک دستور، آن دستور نامعتبر تلقی می‌شود.
- آسیب رسانی به قهرمانان خودی وجود ندارد و قابلیت‌های خطی تهاجمی از قهرمانان خودی عبور می‌کند.

محدودیت‌های نقشه‌ها

- ابعاد نقشه همواره ۳۱*۳۱ است.
- خانه‌های معمولی نقشه همواره همبند است. به این معنی که از هر خانه با حرکت عادی می‌توان به هر خانه‌ی دیگر رفت.
- ابعاد منطقه‌ی هدف بین یک مربع ۵*۵ تا یک مربع ۹*۹ متغیر است. لزومی ندارد که این منطقه مستطیل شکل باشد.
- خانه‌های منطقه‌ی هدف کاملا همبند است.
- فاصله‌ی منطقه‌های زنده‌شدن دو تیم از منطقه‌ی هدف برابر است.
- منطقه‌ی زنده‌شدن هر تیم تنها ۴ خانه‌ی معمولیست و این ۴ خانه همیشه در نزدیکی هم قرار دارند.

مشخصات منابع تخصیص یافته به هر کد

- حافظه اصلی (RAM): ۱ گیگابایت
- پردازنده (CPU): ۱ هسته
- حافظه دائمی (HDD Storage): ۲۰۰ مگابایت

دستورات کمکی محیط Command Line:

- با دستور زیر می‌توانید از طریق Command Line نقشه‌ی مورد نظر خود را به عنوان ورودی پیش از اجرا به سرور بدهید:

```
export AICMap=/path/to/map
```

- با آپشن extra=400– می‌توان به محدودیت زمانی‌های بازی ۴۰۰ میلی ثانیه اضافه کرد. زیرا به طور معمول سیستم‌های کاربران کمی کندتر از محیط اصلی اجرای بازی عمل می‌کنند.
- با آپشن view– می‌توانید فایل html که نمایشگر دوبعدی بازی می‌باشد را برای مشاهده‌ی بازی دریافت کنید.

شروع کد نویسی (بایدها و نبایدها)

۱.شما باید کد هوش مصنوعی خود را در توابع preProcessTurn و pickTurn و moveTurn و actionTurn در فایل AI.java (برای زبان‌های دیگر فایلی به همین نام) قرار دهید.

۲. شما می‌توانید کد کلاینت داده شده را تغییر دهید، به آن فایل اضافه کنید یا از آن فایل حذف کنید، به شرط آن که تغییرات داده شده در کامپایل و اجرای کلاینت و ارتباط آن با سرور اختلالی ایجاد نکند. در مورد هر کلاینت نکاتی ذکر شده که به آن‌ها نیز باید توجه شود. هم‌چنین باید تغییرات احتمالی فایل‌های دیگر کلاینت (فایل‌هایی غیر از فایلی که در آن کد می‌زنید) را در نظر بگیرید.

۳. شما می‌توانید برای به روز بودن کلاینت‌ها یا سرور خود به آخرین نسخه منتشر شده در repository مسابقات مراجعه کنید. این کار کاملا اختیاری است و اگر به کار با git آشنایی ندارید توصیه نمی‌شود.

https://github.com/SharifAIChallenge/

(repo کلاینت‌ها با پیشوند AIC19-Client آغاز می‌شود)

نحوه‌ی اجرا و توسعه (کلاینت جاوا)

پیشنهاد ما برای توسعه‌ی کلاینت جاوا استفاده از IntelliJ است. پس از بازکردن IntelliJ گزینه‌ی Import Project را انتخاب کنید و پوشه‌ای که کلاینت در آن قرار دارد را انتخاب کنید. در صفحه‌ی Libraries مطمئن شوید gson-2-3-1.jar انتخاب شده است.

نحوه‌ی اجرا و توسعه (کلاینت سی‌پلاس‌پلاس)

برای توسعه‌ی کلاینت سی‌پلاس‌پلاس پیشنهاد ما استفاده از Clion است. پس از نصب Clion در صورتی که از ویندوز استفاده می‌کنید از توضیحات لینک زیر برای تنظیم Clion استفاده کنید:

https://www.jetbrains.com/help/clion/quick-tutorial-on-configuring-clion-on-windows.html

پس از نصب و راه‌اندازی Clion کافیست پوشه‌ای که پروژه در آن قرار دارد را در Clion باز کنید.

در صورتی که نمی‌خواهید از Clion استفاده کنید نیاز به نصب Cmake دارید. پس از نصب Cmake یک ترمینال باز کنید و به پوشه‌ای که پروژه در آن قرار دارد بروید و دستور Cmake را اجرا کنید. پس از این می‌توانید با استفاده از دستور make پروژه را build کنید. بعد از اجرای دستور make فایل باینری client در پوشه‌ی جاری ساخته خواهد شد.

نحوه‌ی اجرا و توسعه (کلاینت پایتون)

برای اجرای کلاینت پایتون نیاز به پایتون ۳ دارید. پس از نصب پایتون ۳ کافیست فایل Controller.py را اجرا نمایید. شما می‌توانید به این کلاینت فایل‌های مورد نظرتان را اضافه کنید یا فایل‌های آن را تغییر دهید به شرط آن که کد شما با سرور سازگار باشد . به سادگی بتوانید نسخه‌های احتمالی بعدی کلاینت را جایگزین نمایید. برای توسعه و کدنویسی به این زبان، PyCharm را توصیه می‌کنیم.

ارسال فایل ها

برای ارسال فایل‌های خود، پس از اعمال تغییرات دلخواه خود ، تمامی محتویات دریافت شده از ریپازیتوری کلاینت را فشرده کنید و فایل فشرده (با پسوند zip) را ارسال نمایید.

رابط برنامه نویسی:

نکات مهم:

۱.رفرنس‌های موجود در نقشه، در هر نوبت از ابتدا ساخته می‌شوند و رفرنس‌های نوبت قبل معتبر نیستند. (برای حفظ دسترسی به آبجکت‌ها، توصیه می‌شود از id آن‌ها، که همواره ثابت است، استفاده کنید.)

۲. در تمامی قسمت هایی که علامت ستاره (*) وجود دارد، تمامی حالات تابع شامل شناسه‌ی قهرمان به جای آبجکت قهرمان، مختصات سطر و ستون به جای آبجکت خانه و نام قابلیت به جای آبجکت قابلیت پیاده شده است.

۳. در کلاینت پایتون، متدهای get و set پیاده نشده‌اند و به متغیرها دسترسی مستقیم وجود دارد.

۴. در کلاینت پایتون، متد های ذکر شده در مورد شماره ۲، به صورت دیگری مقداردهی می‌شوند. برای مثال فرض کنید در یک جا Cell و در جای دیگر row و column به عنوان ورودی داده شود. نحوه نگارش به صورت زیر خواهد بود:

method_name(Cell = ..., ...) **method_name(row = ..., column = ..., ...)**

در ادامه، اجزایی که برای برنامه‌نویسی به آنان نیاز دارید به همراه توضیحی مختصر آمده‌اند. سینتکس این توابع متعلق به کلاینت جاوا می‌باشد. برای کلاینت پایتون همین اسامی به صورت snake_case می‌باشند.

Class (or interface) Ability	
`int getRemCooldown();`	مقدار باقی‌مانده از مدت زمان غیر فعال. این مقدار برای قابلیت‌های دشمن نامعتبر است.
`boolean isReady();`	در صورتی که قابلیت آماده‌ی انجام در این نوبت باشد، true برمی‌گرداند. این مقدار برای قابلیت‌های دشمن نامعتبر است.
همه‌ی متدهای کلاس AbilityConstants در این کلاس نیز موجود است.	

Class (or interface) AbilityConstants	
`AbilityName getName();`	نام قابلیت
`AbilityType getType();`	نوع قابلیت
`int getRange();`	حداکثر فاصله‌ی اعمال
`int getAPCost();`	هزینه‌ی انجام قابلیت
`int getCooldown();`	مدت زمان غیر فعال بودن قابلیت پس از استفاده
`int getAreaOfEffect();`	برد اثرگذاری قابلیت
`int getPower();`	قدرت قابلیت

`boolean isLobbing();`	پرتابی/خطّی بودن قابلیت. برای پرتابی مقدار true و برای خطّی مقدار false.
`bool isPiercing();`	شکافندگی قابلیت. در صورت شکافنده بودن true برمی‌گرداند و در غیر این‌صورت false.

`enum AbilityName`	
`SENTRY_ATTACK`	حمله‌ی قراول
`SENTRY_DODGE`	گریز قراول
`SENTRY_RAY`	اشعه‌ی قراول
`BLASTER_ATTACK`	حمله‌ی ویرانگر
`BLASTER_DODGE`	گریز ویرانگر
`BLASTER_BOMB`	بمب ویرانگر
`HEALER_ATTACK`	حمله‌ی درمانگر
`HEALER_DODGE`	گریز درمانگر
`HEALER_HEAL`	درمانگری درمانگر
`GUARDIAN_ATTACK`	حمله‌ی محافظ
`GUARDIAN_DODGE`	گریز محافظ
`GUARDIAN_FORTIFY`	تقویت محافظ
`SHADOW_ATTACK`	حمله‌ی سایه
`SHADOW_DODGE`	گریز سایه
`SHADOW_SLASH`	برش سایه

`enum AbilityType`	
`DEFENSIVE`	دفاعی
`DODGE`	گریز
`OFFENSIVE`	تهاجمی

`Class (or interface) CastAbility`	
`int getCasterId();`	شناسه‌ی قهرمان اعمال کننده‌ی قابلیت. در صورت عدم مشاهده‌ی قهرمان اعمال کننده، نامعتبر است.
`int[] getTargetHeroIds();`	شناسه‌های قهرمان‌های تحت تاثیر قابلیت که در دید هستند. برای گریز خود قهرمان جزء تحت تاثیرها به حساب نمی‌آید.
`Cell getStartCell();`	خانه‌ی مبدا اعمال قابلیت. در صورت عدم حضور این خانه در دید، نامعتبر است.

`Cell getEndCell();`	خانه‌ی مقصد اعمال قابلیت. در صورت عدم حضور این خانه در دید، نامعتبر است.
`AbilityName getAbilityName();`	نام قابلیت. در صورت نامعتبر بودن همه اطلاعات بالا، نامعتبر است.

Class (or interface) Cell	
`int getRow();`	شماره‌ی سطر خانه
`int getColumn();`	شماره‌ی ستون خانه
`boolean isWall();`	در صورت مسدود بودن خانه true و در صورت معمولی بودن false برمی‌گرداند.
`boolean isInMyRespawnZone();`	اگر در منطقه‌ی زنده شدن خودی باشد، true برمی‌گرداند.
`boolean isInOppRespawnZone();`	اگر در منطقه‌ی زنده شدن دشمن باشد، true برمی‌گرداند.
`boolean isInObjectiveZone();`	اگر در منطقه‌ی هدف باشد، true برمی‌گرداند.
`boolean isInVision();`	اگر در محدوده‌ی دید فعلی باشد، true برمی‌گرداند.

enum Direction	
`UP`	بالا
`DOWN`	پایین
`LEFT`	چپ
`RIGHT`	راست

Class (or interface) Hero	
`Ability getAbility(AbilityName);`	قابلیت با این نام را در صورت وجود برمی‌گرداند.
`int getId();`	شناسه‌ی قهرمان را برمی‌گرداند.
`Ability[] getAbilities();`	قابلیت‌های قهرمان را برمی‌گراند.
`Ability[] getDefensiveAbilities();`	قابلیت‌های دفاعی قهرمان را برمی‌گرداند.
`Ability[] getDodgeAbilities();`	قابلیت‌های گریز قهرمان را برمی‌گرداند. اندازه‌ی این آرایه همواره یک است.
`Ability[] getOffensiveAbilities();`	قابلیت‌های تهاجمی قهرمان را برمی‌گرداند.
`int getCurrentHP();`	سلامتی کنونی قهرمان. برای قهرمانان دشمن خارج از دید، نامعتبر است.
`Cell getCurrentCell();`	خانه‌ی کنونی قهرمان. برای قهرمانان دشمن خارج از دید، نامعتبر است.

`Cell[] getRecentPath();`	مسیر حرکت قهرمان در فاز پیشین. برای قهرمانان دشمن، حرکت هایی که در محدوده‌ی دید بوده برمی‌گردد.
`int getRemRespawnTime();`	مقدار باقی‌مانده از مدت زمان زنده شدن. برای قهرمانان دشمن نیز همواره معتبر است.
همه‌ی متدهای کلاس HeroConstants در این کلاس نیز موجود است.	

Class (or interface) HeroConstants	
`HeroName getName();`	نوع قهرمان را برمی‌گرداند.
`AbilityName[] getAbilityNames();`	نام قابلیت‌های قهرمان را برمی‌گرداند.
`int getMaxHP();`	سلامتی اولیه قهرمان
`int getMoveAPCost();`	هزینه حرکت قهرمان
`int getRespawnTime();`	مدت زمان زنده شدن

enum HeroName	
`SENTRY`	قراول
`BLASTER`	ویرانگر
`HEALER`	درمانگر
`GUARDIAN`	محافظ
`SHADOW`	سایه

enum Phase	
`PICK`	فاز انتخاب
`MOVE`	فاز حرکت
`ACTION`	فاز عمل

Class (or interface) Map	
`Cell[][] getCells();`	آرایه‌ی دوبعدی خانه‌های نقشه را برمی‌گرداند.
`Cell getCell(int, int);`	خانه‌ی مورد نظر در مختصات داده شده را برمی‌گرداند.
`boolean isInMap(int, int);`	در صورت خارج محدوده بودن مختصات، false برمی‌گرداند.
`int getRowNum();`	تعداد سطرهای نقشه
`int getColumnNum();`	تعداد ستون‌های نقشه
`Cell[] getMyRespawnZone();`	خانه‌های محدوده‌ی زنده شدن خودی را برمی‌گرداند.
`Cell[] getOppRespawnZone();`	خانه‌های محدوده‌ی زنده شدن دشمن را برمی‌گرداند.
`Cell[] getObjectiveZone();`	خانه‌های محدوده‌ی هدف را برمی‌گرداند.

Class (or interface) World	
`Hero getHero(int);`	قهرمان با شناسه‌ی داده شده را برمی‌گرداند.
`Hero getMyHero(Cell);` `Hero getMyHero(int, int);`	قهرمان خودی در خانه‌ی داده شده (مختصات) را برمی‌گرداند.
`Hero getOppHero(Cell);` `Hero getOppHero(int, int);`	قهرمان دشمن را در خانه‌ی داده شده (مختصات) در صورتی که در دید باشد، برمی‌گرداند.
`void castAbility(Hero, Ability, Cell);`	اعمال قابلیت قهرمان به خانه‌ی موردنظر.*
`void moveHero(Hero, Direction);` `void moveHero(int, Direction);`	حرکت قهرمان در جهت مورد نظر به اندازه‌ی یک خانه.
`void pickHero(HeroName);`	انتخاب یکی از این نوع قهرمان در فاز انتخاب.
`Direction[] getPathMoveDirections(Cell, Cell);`	دنباله‌ی جهت ها برای رسیدن از خانه‌ی اول به خانه‌ی دوم.*
`int manhattanDistance(Cell, Cell);`	فاصله منهتنی دو خانه. پیاده‌سازی با مختصات نیز موجود است.*
`Cell getImpactCell(Ability, Cell, Cell);`	در صورت استفاده از قابلیت خطّی، با استفاده از این تابع می‌توان خانه‌ی محل اثر را بر اساس دید فعلی بدست آورد.*
`Hero[] getAbilityTargets(Ability, Cell, Cell);`	در صورت اعمال قابلیت از خانه‌ی اول به دوم، قهرمانانی که مورد اثر قرار می‌گیرند و در دید هستند را برمی‌گرداند.*
`boolean isInVision(Cell, Cell);`	در صورت قابلیت دیدن خانه‌ی دوم از خانه‌ی اول، true برمی‌گرداند.*
`Hero[] getMyHeroes();`	قهرمانان خودی را برمی‌گرداند.
`Hero[] getOppHeroes();`	همه‌ی قهرمانان انتخاب شده‌ی دشمن تا اینجا را برمی‌گرداند. در صورت خارج از محدوده دید بودن، مختصات خانه -1 خواهد بود.
`Hero[] getMyDeadHeroes();`	قهرمانان مرده‌ی خودی را برمی‌گرداند.
`Hero[] getOppDeadHeroes();`	قهرمانان مرده‌ی دشمن را برمی‌گرداند.
`Map getMap();`	نقشه‌ی بازی را برمی‌گرداند.
`CastAbility[] getMyCastAbilities();`	قابلیت های اعمال شده‌ی خودی در نوبت قبلی.
`CastAbility[] getOppCastAbilities();`	قابلیت های اعمال و مشاهده شده‌ی دشمن در نوبت قبلی.
`HeroConstants[] getHeroConstants();`	ویژگی‌های انواع مختلف قهرمانان را برمی‌گرداند.
`AbilityConstants[] getAbilityConstants();`	ویژگی‌های انواع مختلف قابلیت‌ها را برمی‌گرداند.
`int getAP();`	مقدار امتیاز عمل فعلی را برمی‌گرداند.
`int getMyScore();`	امتیاز فعلی خودی را برمی‌گرداند.

`int getOppScore();`	امتیاز فعلی دشمن را برمی‌گرداند.
`int getCurrentTurn();`	شماره نوبت فعلی را برمی‌گرداند.
`Phase getCurrentPhase();`	فاز فعلی را برمی‌گرداند.
`int getMaxScore();`	مقدار سقف امتیاز بازی برای پیروزی را برمی‌گرداند.
`int getMaxAP();`	مقدار اولیه‌ی امتیاز عمل را برمی‌گرداند.
`int getMaxTurns();`	تعداد کل نوبت‌های بازی را برمی‌گرداند.
`int getKillScore();`	امتیاز حاصل از کشتن یک قهرمان دشمن را برمی‌گرداند.
`int getObjectiveZoneScore();`	امتیاز حاصل از حضور یک قهرمان خودی در منطقه‌ی هدف را برمی‌گرداند.
`int getMovePhaseNum();`	شماره‌ی فاز حرکت فعلی در نوبت جاری را برمی‌گرداند.
`int getTotalMovePhases();`	تعداد فازهای حرکت در هر نوبت را برمی‌گرداند.
`int getInitOvertime();`	مقدار اولیه‌ی حداکثر وقت اضافه را برمی‌گرداند.
`int getMaxOvertime();`	مقدار فعلی حداکثر وقت اضافه را برمی‌گرداند.
`int getRemainingOvertime();`	مقدار نوبت باقی مانده تا پایان وقت اضافه فعلی را برمی‌گرداند.
`int getMaxScoreDiff();`	مقدار حداکثر اختلاف امتیاز برای پیروزی را برمی‌گرداند.

Class (or interface) AI	
`void preProcess(World);`	این تابع فقط در ابتدای شروع بازی و پیش از فاز انتخاب اول صدا زده می‌شود. محدودیت زمانی ۵ ثانیه
`void pickTurn(World);`	تابع فازهای انتخاب. به ازای هرکدام از این فازها، یکبار صدا زده می‌شود. محدودیت زمانی ۰٫۷ ثانیه.
`void moveTurn(World);`	تابع فازهای حرکت. به ازای هرکدام از این فازها، یکبار صدا زده می‌شود. محدودیت زمانی اولین فاز هر نوبت، ۱ ثانیه و بقیه ۰٫۷ ثانیه
`void actionTurn(World);`	تابع فازهای عمل. به ازای هرکدام از این فازها، یکبار صدا زده می‌شود. محدودیت زمانی ۰٫۷ ثانیه.