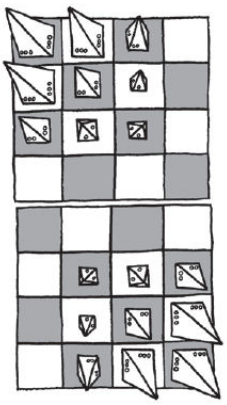
Mesterséges Intelligencia I.
kötelező program

1. Feladat kiírás

A feladat egy Martian Chess játékot játszó ágens írása. A játékot 2 játékos játssza egymás ellen, egy 8x4 méretű sakktáblán. A Martian Chess játékban a bábuk színe nem számít, csupán azok elhelyezkedése, egy adott bábut az birtokolja, akinek a térfelén található. Összesen 3 bábu típust különböztetünk meg: gyalog, drón, királynő, amely bábuk 1, 2 illetve 3 pontot érnek. Minden játékos csak a saját térfelén található bábukkal léphet és csak az ellenfél térfelén található bábukat ütheti le, az ütésért pedig megkapja a leütött bábu pontértékét, természetesen ütés esetén a bábu amivel ütött utána az ellenfélé lesz.A játék akkor ér véget, ha az egyik játékos térfele kiürül, a játékot pedig az nyeri, akinek a legtöbb pontja van. Azonos pontszám esetén az a játékos nyer, aki utoljára lépett (kap +1 pontot).
Játékszabályok:

- Egy körben csak 1 bábut lehet mozgatni, a két játékos felváltva lép.
- A gyalog átlósan léphet 1 mezőt.
- A drón vízszintesen vagy függőlegesen léphet maximum 2 mezőt.
- A királynő egyenes vonalban vagy átlósan léphet tetszőleges mezőnyit (mint sakkban).
- Bábut nem lehet átugrani.
- Ütés akkor történik, ha egy saját bábuval az ellenfél bábuja által elfoglalt mezőre lépünk.
- Ütni nem kötelező.
- Amennyiben egy játékosnak nincs több királynője, készíthet egy újat egy drón és egy gyalog összeolvasztásával, a két bábu közül az egyikkel a másik mezőjére lépve. Hasonló módon drónt is lehet készíteni, ha már nincs több drón a térfelünkön, két gyalog összevonásával.
- Nincs azonnali visszalépés, azaz nem lehet az ellenfél utolsó lépését visszacsinálni (a bábut pont oda visszaléptetni mint ahonnan indult), kivéve akkor, ha az ellenfél leütött valamit.

Kezdetben a két játékos bábui a pálya ellentétes sarkában helyezkednek el, minden bábutípusból 3 db-ot kap minden játékos.



1. ábra. A kezdőállás

2. A keretrendszer

A megoldást Java programozási nyelven kell elkészíteni. A könnyebb fejlesztés és kiértékelés megvalósítása érdekében közreadunk egy keretrendszert, ami egyben egy osztálykönyvtár is. Az említett keretrendszer letölthető az előadás honlapjáról `http://www.inf.u-szeged.hu/~jelasity/mi1/2018/index.html`. A következő fejezetben összefoglaljuk a futtatással, fejlesztéssel és a kiértékeléssel kapcsolatos tudnivalókat.

2.1. Egy meccs futtatása

A keretrendszer egyik alapvető feladata, hogy levezényelje az egyes meccseket (ellenőrizze a lépések helyességét, az időkorlátok betartását, stb...). Ennek technikai megvalósulását szemlélteti a következő példa (linux rendszer alatt, a $ jel a prompt-ot szimbolizálja, egy sorban):

```
$ java -jar game_engine.jar 1 game.mc.MCGame 1234567890 5000
game.mc.players.GreedyPlayer game.mc.players.HumanPlayer
```

Ebben a példában a mohó stratégiát követő példa játékos (GreedyPlayer) játszik az emberi játékos (HumanPlayer) ellen a grafikus felületen. (Mindkét implementáció elérhető a közreadott csomagban.) Az első paraméter (1) a grafikus felületet paraméterezi, 0 esetén nincs grafikus felület, $x \geq 1$ esetben pedig $\frac{1}{x}$ fps-el jeleníti meg a felületet és konzolon bővebb információt kaphatunk a játék során. A harmadik paraméter a játék elején beállított véletlen seed értéke. A negyedik paraméter a gondolkodási időt korlátozza (5000 ms).
A kimenet értelmezésében a következő 2 sor a sarkalatos:

```
0 game.mc.players.GreedyPlayer 0 3.0 5000
1 game.mc.players.HumanPlayer 1 13.0 9223372036854775807
```

ahol mindkét sorban az utolsó előtti szám határozza meg, hogy melyik játékosnak mennyi pontot sikerült elérnie a játék végén. Érvénytelen lépés esetén a játékos -1 ponttal veszít.

2.2. Fejlesztés

A közreadott keretrendszer egyben egy osztálykönyvtár is, hiszen tartalmazza azt az absztrakt osztályt (`game.mc.MCPlayer`), amelyet a megoldásoknak ki kell terjeszteniük.
A könnyebb megértés kedvéért tekintsünk egy példát a fejlesztés-kiértékelés ciklusra:

1. Töltsük le és tanulmányozzuk a közreadott keretrendszert, valamint a benne elérhető `game.mc.MCPlayer` absztrakt osztályt. Értsük meg a meccs lejátszásának mikéntjét az említett osztály szemszögéből. Ehhez nagy segítség a javadoc-ban elérhető dokumentáció.

2. Implementáljuk a stratégiánkat Java programozási nyelven. Például:

```
import java.util.List;
import java.util.Random;

import game.engine.utils.Pair;
import game.mc.MCAction;
import game.mc.MCPlayer;

public class h241696 extends MCPlayer {
  public h241696(int color, int[][] board, Random r) {
    super(color, board, r);
  }
```

```
  @Override
  public MCAction getAction(List<Pair<Integer, MCAction>> prevAction) {
    return new MCAction(r.nextInt(board.length), r.nextInt(board[0].length),
    r.nextInt(board.length), r.nextInt(board[0].length));
  }

}
```

Fontos: Ne tegyük csomagba (`package`) a megoldásunkat!

Fontos: A megoldást tartalmazó publikus osztályunk neve legyen a h-s azonosítónk!

3. Fordítsuk le a kódunkat. A fordítás során a class-path-ban szerepelnie kell a keretrendszernek. A fordítás a következő módon tehető meg:

   ```
   $ javac -cp game_engine.jar h241696.java
   ```

4. Az elkészített stratégiát próbáljuk ki, valamelyik közreadott stratégia ellen a keretrendszer használatával. Például (egy sorban):

   ```
   $ java -jar game_engine.jar 0 game.mc.MCGame 1234567890 5000
   game.mc.players.GreedyPlayer h241696
   ```

   Érdemes több tesztet végezni, különböző random seed-el, illetve a játékosok sorrendjének felcserélésével. (**Windows rendszeren kettőspont helyett pontosvessző használandó.**)

5. Ha megfelelőnek ítéljük a stratégiánkat, akkor töltsük fel a Bíró rendszerbe, különben iteráljunk a 2. ponttól.

2.3. Követelmények a megoldással szemben

A megoldást tartalmazó forráskódnak minden körülmények között ki kell elégítenie a következő követelményeket:

- A feltöltött forráskódnak le kell fordulnia
- Segédosztály írása engedélyezett, de minden kódnak egy fájlba kell kerülnie.
- A stratégiánkat tartalmazó osztálynak a `game.mc.MCPlayer`-ból kell származnia, ami a keretrendszer részét képezi.
- A kiértékeléshez az `game.engine.Engine` osztályban található main-t fogjuk használni. Ez egy meccset játszik le (l. dokumentáció és forráskód).
- A `GreedyPlayer`-t le kell győzni a következő arányban: 10 játékból legalább 8-szor kell nyerni, és a 10 játékból 5-ször az egyik, 5-ször a másik játékos kezd. Összesen tehát 10 parti van. Partinként 5000 ms gondolkodási idő áll majd rendelkezésre, amennyiben ebből kifut az algoritmus automatikusan veszít. Ha sikerül legalább 8 győzelmet elérni, akkor a 12 pont megvan.
- Érvénytelen lépés estén az algoritmus automatikusan veszít.
- A sikeres játékosok folyamatosan tornákon vesznek részt november 2-től. Javított változatok feltöltésére is lehetőség lesz december 3-ig. A kialakult végső rangsor fogja a végső pontszámokat meghatározni.

- A megoldást tartalmazó osztálynak részletes magyar osztálydokumentációt kell tartalmaznia, javadoc formátumban, illetve a kód dokumentációja is magyar kell, hogy legyen.
- A kód nem használhat a keretrendszeren kívül semmilyen más osztálykönyvtárat (természetesen a JDK 8 osztályain kívül).
- A stratégiát leíró osztály nevének meg kell egyeznie a hallgató h-s azonosítójával (kis *h* betűvel).
- A megoldást tartalmazó osztály nem lehet csomagban.
- A megoldásban nem lehet képernyőre írás.
- A megoldás nem nyithat meg file-t, az ágensen belül nem indíthat új szálat, kódot nem hajthat végre párhuzamosan.
- Az implementált metódusoknak minden esetben vissza kell térniük. (Nem szerepelhet benne exit hívás például.)
- A forráskód első sorában megadható egy nicknév a következő formában:

  `///Nicknevem`

  Amennyiben ez adott, a hallgató ez alapján azonosíthatja magát a tournament ranglistájában.

Néhány szabályt nagyon szigorúan fogunk venni tekintettel a magas létszámra. A kötelező program nem kerül elfogadásra (ami a kurzuson bukást jelent) ha:

- a benyújtott ágens nem vagy helytelenül öröklődik
- a futás során hiba lép fel (végtelen ciklus, kivételdobás, stb.)
- a forráskódban nincs kielégítő magyar nyelvű osztálydokumentáció és forráskód dokumentáció (azért magyar, hogy nehogy kísértésbe essünk letöltögetésre)
- a programozói munka nem a hallgató saját munkája (természetesen ötleteket lehet és ajánlott keresgélni); ez esetben a bukáson felül további következményekkel is kell számolni! Vannak megbízható eszközeink a plágium kiszűrésére, nem érdemes kísérletezni.
- bármilyen "hack"-et találunk ami a kiértékelő nem rendeltetésszerű használatából fakadóan jogosulatlan előnyt eredményez (néhány példa a CooSpace fórumon).

2.4. Kiértékelés

A kötelező programok tényleges kiértékelését a bíró rendszer végzi. Ha sikerült elkészíteni egy megfelelőnek ítélt stratégiát, akkor a bíró rendszerbe (`https://biro.inf.u-szeged.hu/`) történő bejelentkezés után, fel kell tölteni a forráskódot. A feltöltött forráskódnak ki kell elégíteni az előző fejezetben ismertetett követelményeket!
A feltöltés után a bíró rendszer a következők szerint jár el:

- A feltöltött forráskódon egy előzetes ellenőrzést hajt végre a Bíró, hogy az kielégíti-e az előzőekben ismertetett követelményeket (elnevezés, file méret korlát, stb...). Ha nem elégíti ki azokat, akkor a beküldés sikertelen, máskülönben a kiértékelés folytatódik.
- A rendszer lefordítja a feltöltött forrást. A fordítás a 'Fejlesztés' c. alfejezetben látott példa szerint történik. Ha nem sikerült lefordítani a kódot, akkor a beküldés sikertelen, máskülönben a kiértékelés folytatódik.

- A rendszer 10 alkalommal meghívja a kiértékelő keretrendszert az 'Egy meccs futtatása' c. alfejezetben szereplő példában látottak szerint a `GreedyPlayer` stratégia ellen. A feltöltött megoldás váltakozva kezdő illetve második játékosként szerepel. Ha sikerül a megoldásunknak legalább 8 alkalommal győznie, akkor a kiértékelés tovább halad. Ha hiba történt bármelyik meccs során, akkor a beküldés sikertelen.
- A bíró rendszer elérhetővé teszi a webes felületen az elért pontszámunkat és a kiértékelés során készített jelentést. A pontszám pontosan akkor 1, ha a beküldés nem lett sikertelen a fenti pontok egyikében sem, különben 0.
- A bíró rendszer ütemezi a feltöltést tournament rendszerben való kiértékelésre.

Egy beküldés akkor tekinthető elfogadottnak, ha legalább 1 pontot ad a bíró rendszer.
A következőkben egy példa bíró kimenet látható.

```
Player: h241696
Valid upload!
Enemy: game.mc.players.GreedyPlayer
MatchID: 1, Seed: 32681
0 h241696 0 6.0 109
1 game.mc.players.GreedyPlayer 1 14.0 5000
MatchID: 2, Seed: 22853
0 game.mc.players.GreedyPlayer 0 11.0 5000
1 h241696 1 13.0 197
MatchID: 3, Seed: 30686
0 h241696 0 13.0 117
1 game.mc.players.GreedyPlayer 1 7.0 5000
MatchID: 4, Seed: 18873
0 game.mc.players.GreedyPlayer 0 6.0 5000
1 h241696 1 11.0 198
MatchID: 5, Seed: 9454
0 h241696 0 13.0 103
1 game.mc.players.GreedyPlayer 1 9.0 5000
MatchID: 6, Seed: 7210
0 game.mc.players.GreedyPlayer 0 16.0 5000
1 h241696 1 18.0 196
MatchID: 7, Seed: 18056
0 h241696 0 14.0 119
1 game.mc.players.GreedyPlayer 1 3.0 4999
MatchID: 8, Seed: 18873
0 game.mc.players.GreedyPlayer 0 7.0 4999
1 h241696 1 10.0 197
MatchID: 9, Seed: 14163
0 h241696 0 14.0 121
1 game.mc.players.GreedyPlayer 1 9.0 5000
MatchID: 10, Seed: 25559
0 game.mc.players.GreedyPlayer 0 9.0 5000
1 h241696 1 12.0 196
1
```

Látható, hogy a fenti futás egy sikeres feltöltéshez tartozik, mivel a legutolsó sorban szereplő sor—a Bíró által adott pontszám—1. Az egyes blokkok (Enemy: ... kezdetű soroktól kezdődő szakaszok) az egyes ellenfelek elleni meccsek leírásai. Minden meccset 3 sor ír le. Egy meccsről a következő információk kerülnek kiírásra: meccs sorszáma, véletlen seed értéke, illetve a két

játékos pontszáma a játék végén. A véletlen seed és az ellenfél stratégiájának ismeretében reprodukálható a meccs a keretrendszerben (lásd bővebben: véletlen seed paraméter az 'Egy meccs futtatása' c. alfejezetben). Amennyiben a játékos timeout-ol a reprodukálhatóság nem garantált.
Fontos megjegyezni, hogy bizonyos esetekben a kiértékelés hosszú ideig tart. Ezalatt a bíró rendszer webes felületén semmi információt nem látunk. Ez normális működés! Ha ez alatt frissítjük a böngészőt és újraküldjük az adatokat, akkor az új próbálkozásnak minősül!
Másik fontos észrevétel, hogy a beküldési határidőhöz közeledve egyre terheltebb szokott lenni a rendszer. Ekkor egy-egy feltöltés hosszabb időt is igénybe vehet, fejlesztés során ezzel mindenki kalkuláljon! (Természetesen a rendszer processzor időt mér, így a terheltség nem jelenti azt, hogy kevesebb idő jut az egyes stratégiáknak arra, hogy végigjátsszák a meccset.)

2.5. Tournament rendszer

Minden hallgatónak a bíró által legutoljára elfogadott feltöltése játszik a tournament-ben (az-az legalább olyan megoldás, ami átmegy a forráskód ellenőrzésen, sikeresen lefordul, és legyőzi a baseline ellenfeleket). Ebben a rendszerben az összes említett forrást lefordítjuk és teljes körmérkőzésen egymás ellen játszatjuk. Elérhetősége: `http://rgai.inf.u-szeged.hu/mi1kotprog/`. A beadás lezárulta után a minimum pontszám feletti pontszámok a tournament-ben elért helyezés függvényében fognak kiosztásra kerülni. (Tehát az utolsó sikeres feltöltés számít.) Az összes többi hallgató által feltöltött ágens ellen kell játszani. Minden játékospár fix számú meccset játszik váltakozó kezdéssel, és a nyerések arányában kapnak pontot.

2.6. Korlátok, határidők

Az alábbi korlátok, határidők élnek:

- Egy meccs maximum 10000 ms-ig tart, így egy játékos összesen 5000 ms-el rendelkezik (CPU idő).
- Amennyiben valamelyik játékos gondolkozási ideje elfogy, a játék véget ér.
- A beadott forrás file mérete legfeljebb 200 KB lehet.
- Egy baseline `GreedyPlayer` ellenfél ellen játszott meccs során a teljes keretrendszer számára 1 GB memória áll rendelkezésre.
- A bíró 1.8-os Java verziót használ.
- A bíró szöveges állományait (így a forráskódokat is) UTF-8 (without BOM) kódolás szerint tárolja.
- 50 próbálkozás áll a hallgatók rendelkezésére.
- A végső beküldési határidő 2018. december 3. 23:59.

Hasznos és jó munkát kívánunk!