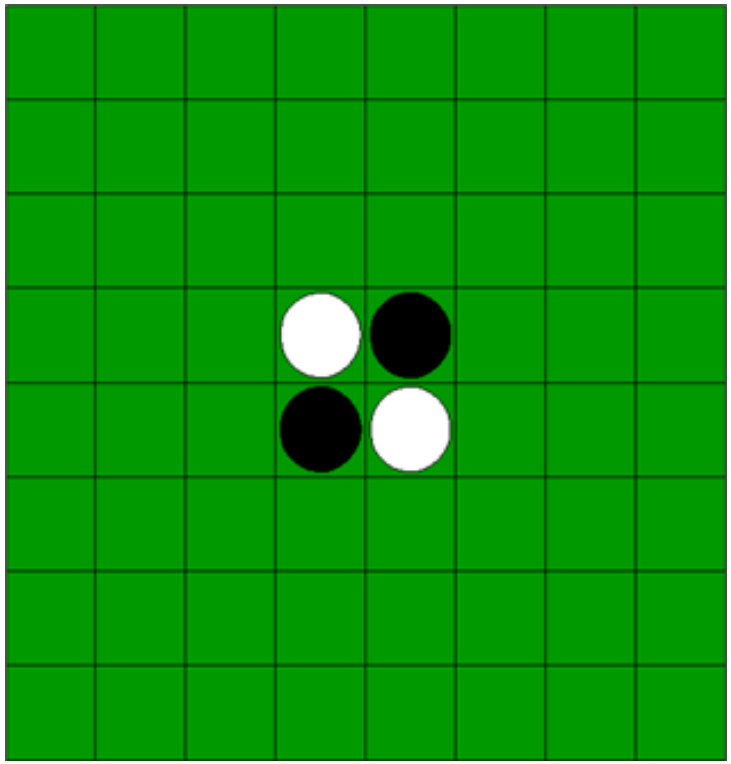
Programmierprojekt: Othello

Spielregeln



Gegeben

- Spielfeld: 8x8
- Steine in unbegrenzter Anzahl

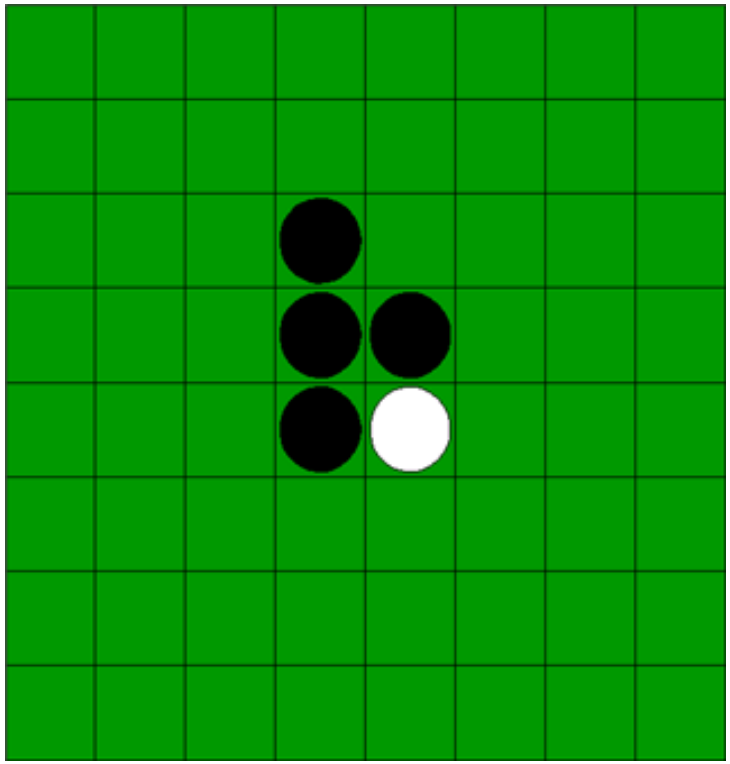
Ziel

- Möglichst viele Felder besetzen

Spielregeln

- Start
  - 4 Steine vorgegeben (2x weiß, 2x schwarz)
  - Schwarz beginnt
- Stein kann gesetzt werden, wenn:
  - Spielfeld ist frei
  - Gegnerisches Stein grenzt direkt an
  - Zug dreht Steine um
- Alle gegnerischen Steine zwischen zwei eigenen (diagonal/vertikal/horizontal) werden umgedreht

Spielregeln



Gegeben

- Spielfeld: 8x8
- Steine in unbegrenzter Anzahl

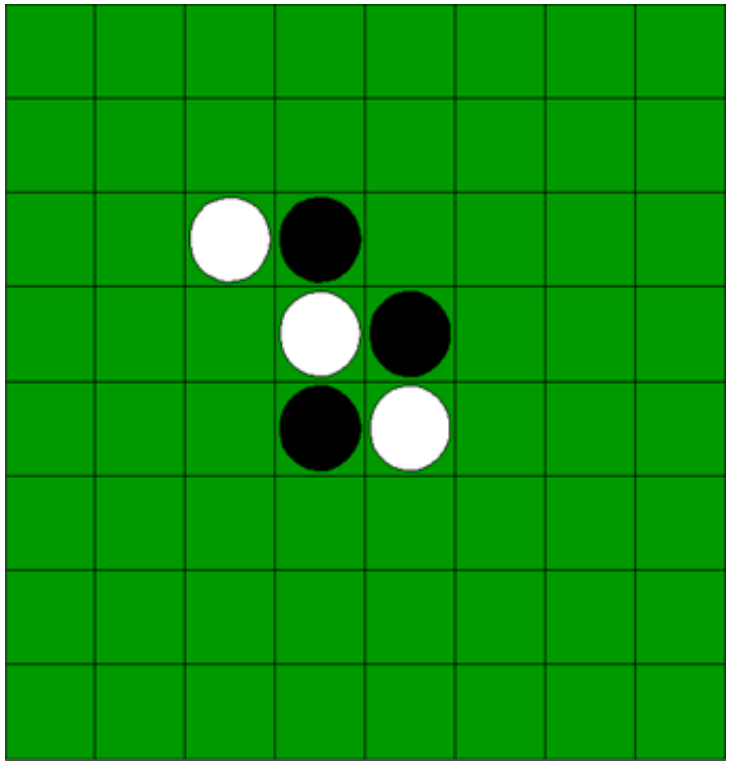
Ziel

- Möglichst viele Felder besetzen

Spielregeln

- Start
  - 4 Steine vorgegeben (2x weiß, 2x schwarz)
  - Schwarz beginnt
- Stein kann gesetzt werden, wenn:
  - Spielfeld ist frei
  - Gegnerisches Stein grenzt direkt an
  - Zug dreht Steine um
- Alle gegnerischen Steine zwischen zwei eigenen (diagonal/vertikal/horizontal) werden umgedreht

Spielregeln



Gegeben

- Spielfeld: 8x8
- Steine in unbegrenzter Anzahl

Ziel

- Möglichst viele Felder besetzen

Spielregeln

- Start
  - 4 Steine vorgegeben (2x weiß, 2x schwarz)
  - Schwarz beginnt
- Stein kann gesetzt werden, wenn:
  - Spielfeld ist frei
  - Gegnerisches Stein grenzt direkt an
  - Zug dreht Steine um
- Alle gegnerischen Steine zwischen zwei eigenen (diagonal/vertikal/horizontal) werden umgedreht

Umsetzung

Spieler-Programm: Ihre Aufgabe

- Hauptfunktion: `Brett_Neu = Spieler_Gruppe_X(Brett_Alt,color,t)`
- `Brett`: 8x8 Matrix mit Brett-Besetzung
  - Brett(i,j) = 0 => Feld (i,j) unbesetzt; 1 := weißer Stein; -1 := schwarzer Stein
- `Color`: Farbe des Spielers
  - +1: weiß
  - - 1: schwarz
- `t`: verbleibende Zeit für alle weiteren Züge

Tournier-Server: wird gestellt

- Aufruf: `tournament_main('time_budget',180,'games_per_pair', 2)`
- Funktion:
  - Ruft Spieler-Programme auf
  - Alle Spieler (*.m-Dateien) aus dem Ordner “players” spielen je `games_per_pair` mal gegeneinander (2 Mal im Endspiel)
  - Prüft Korrektheit der Züge (Spieler verliert nach ungültigem Zug)
  - Erzeugt Logfiles mit Spielverlauf
- 3 sehr einfache Spieler mitgeliefert

Verfügbar unter Moodle: https://www.moodle.tum.de

Regeln

- Reine Matlab-Implementierung
  - erlaubt: `*.m, *.mat` - Dateien
  - Nicht erlaubt: Einbindung von C/C++, Java, Fortran etc.
- Alle Berechnungen im selben Matlab-Prozess. Nicht erlaubt sind:
  - Erzeugen weiterer Prozesse
  - Aufbau von Netzwerkverbindungen zu anderen Programmen
- Code muss ohne Anpassungen auf anderem Rechner lauffähig sein (Pfad!)
- Algorithmen: Minimax/eine Variante davon: http://de.wikipedia.org/wiki/Minimax-Algorithmus

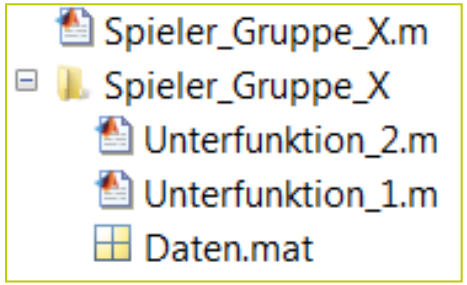
Regeln

Spielregeln

- Jeder Spieler hat ein Zeitbudget von 180 Sekunden, die er für alle Züge aufbrauchen darf. Braucht er z.B. 10 s für den ersten Zug, bleiben für die restlichen Züge noch 170 s.
- Spieler, der sein Zeitbudget aufbraucht, wird mit 0 Punkten disqualifiziert

Abgabe

- Vorgeschriebene Ordnerstruktur
  - Hauptfunktion `<Spieler-Name>.m`
  - Weitere Funktionen im Verzeichnis `<Spieler-Name>`
  - **Kreative Spieler-Namen sind willkommen!**
- Gesamtgröße aller Dateien darf 2MB nicht überschreiten



Kalibrierung

- Zeitbedarf für Zug-Berechnung bis Tiefe N Hardwareabhängig
- Sie dürfen *ein Mal* eine Kalibrierungsfunktion einreichen, um die Hardware des Turnier-Rechners zu testen
- Anforderungen
  - Eine einzige m-Datei wird eingereicht
  - Datei muss bei direktem Aufruf lauffähig sein (Pfad!)
  - Abbruch nach 5 Minuten
  - Datei muss eine Ausgabedatei im txt/mat-Format generieren
    - Diese wird Ihnen zurückgeschickt
- Zweck ist Anpassung ihres Spielers an Hardware
  - Programmieren Sie einen Benchmark
  - Einsenden eines Spieler nicht untersagt, aber nicht zweckdienlich

Hausaufgabe

- Bilden Sie eine Gruppe in Moodle
  Zeitslots:

  15:00 - 15:20 | Gruppe 1 | Gruppe 2
  15:20 - 15:40 | Gruppe 3 | Gruppe 4
  15:40 - 16:00 | Gruppe 5 | Gruppe 6
  16:00 - 16:20 | Gruppe 7 | Gruppe 8

- Starten Sie den Turnierserver
- Planen Sie: welche Unterfunktionen werden benötigt?
- Teilen Sie die Aufgaben auf