Inleiding Programmeren 2016-2017
Huiswerk 4: Schaken

toon.calders@uantwerpen.be

Belangrijke Informatie

- Maak deze opdracht **individueel**;
- Dien de opdracht in op Blackboard, onder "opdrachten";
- De deadline is woensdag 9/11/2016 om 23:59 (volgens de klok van Blackboard!);
- Benodigdheden: Python 3;

Omschrijving

In deze opdracht maak je een schaakspel dat door twee personen gespeeld kan worden. Het schaakspel laat enkel geldige zetten toe en test op schaakmat. De opdracht is opgedeeld in onderdelen van stijgende complexiteit. Afhankelijk van welke onderdelen correct geïmplementeerd werden, wordt het uiteindelijke resultaat voor deze opdracht bepaald. Het is dus niet nodig om alle onderdelen te implementeren om te slagen voor deze opdracht.

Opdracht

De regels van het schaakspel vind je op:

`http://www.schaakzone.nl/schaken-voor-beginners/spelregels-van-schaken.php`.

Negeer de bijzondere spelregels "en passant", "rokade" en "promotie".

Mocht je niet vertrouwd zijn met schaken, aarzel dan niet om een van de assistenten aan te spreken voor een korte introductie; deze opdracht heeft als doel jouw programmeervaardigheden te testen, niet jouw kennis van het schaakspel!

Zet het bord maar klaar

Implementeer een klasse "Schaakstuk" met subklassen "Pion", "Toren", "Paard", "Loper", "Koningin", en "Koning". Voor deze schaakstukken moeten volgende methodes correct geïmplementeerd worden:

- Constructor met parameter "kleur". Nieuwe schaakstukken aanmaken gaat dus als volgt:

```
# Definitie van twee constanten:
WIT=1
ZWART=2

# Aanmaken stukken:
p=Pion(WIT)
t=Toren(ZWART)
```

- Overload method `__str__` geeft informatie over het schaakstuk zoals het type en de kleur:

```
print(p)    # Output: Witte pion.
print(t)    # Output: Zwarte toren.
```

- Methode "get_kleur" die de kleur van een schaakstuk geeft:

```
print(p.get_kleur(),t.get_kleur())    # Output: 1 2
```

Implementeer een klasse "Schaakbord". Deze klasse moet volgende methodes correct implementeren:

- Methodes "plaats(stuk,(rij,kolom))", "verwijder(stuk)" en "verplaats(stuk,(rij,kolom))" om schaakstukken toe te voegen op het bord, weg te halen van het bord, en te verplaatsen. "waar_is(stuk)" geeft de positie in de vorm van een tupel (rij,kolom) en stuk_op((rij,kolom)) geeft het stuk dat op positie (rij,kolom) staat. Deze beide methodes geven 'None' terug in het geval "stuk" niet op het bord staat, respectievelijk er geen stuk op de positie (rij,kolom) staat.

```
S=Schaakbord()
S.plaats(p,(2,1))
S.plaats(t,(8,1))
print(S.waar_is(t))        # Output: (8,1)
S.verplaats(p,(4,1))
print(S.stuk_op((2,1)))    # Output: None
print(S.stuk_op((4,1)))    # Output: Witte pion.
```

- Overload method `__str__` geeft het schaakbord weer. `print(S)` geeft volgende (of vergelijkbare) output:

```
   a  b  c  d  e  f  g  h

8   Rb .  .  .  .  .  .  .
7   .  .  .  .  .  .  .  .
6   .  .  .  .  .  .  .  .
5   .  .  .  .  .  .  .  .
4   Pw .  .  .  .  .  .  .
3   .  .  .  .  .  .  .  .
2   .  .  .  .  .  .  .  .
1   .  .  .  .  .  .  .  .
```

Gebruik voor de stukken twee karakters; het eerste karakter geeft het type van het stuk weer: R(ook) voor toren, (k)N(ight) voor paard, B(ishop) voor loper, Q(ueen) voor koningin, K(ing) voor koning, en P(awn) voor pion. Het tweede karakter duidt de kleur van het stuk aan; w(hite) of b(lack).

- Methode "zet_begin_positie()" maakt alle benodigde stukken aan voor een schaakspel en plaatst ze op de juiste posities. Dus, `S.zet_begin_positie()` stelt het bord als volgt op:

```
   a  b  c  d  e  f  g  h

8   Rb Nb Bb Qb Kb Bb Nb Rb
7   Pb Pb Pb Pb Pb Pb Pb Pb
6   .  .  .  .  .  .  .  .
5   .  .  .  .  .  .  .  .
4   .  .  .  .  .  .  .  .
3   .  .  .  .  .  .  .  .
2   Pw Pw Pw Pw Pw Pw Pw Pw
1   Rw Nw Bw Qw Kw Bw Nw Rw
```

- Voorzie een constructor indien nodig.

Denk op voorhand goed na welke datastructuren je gebruikt; een goed gekozen data structuur kan jouw programma aanzienlijk vereenvoudigen!

A game of Pawns

Nu het bord klaar staat, is het tijd om te beginnen spelen. We starten met de bewegingen van de pion. Je hoeft in deze fase geen rekening te houden met "schaak" en "schaakmat".

- Voeg een methode `geldige_zetten(Schaakbord)` toe aan de klassen Schaakstuk en Pion. Deze methode geeft een lijst met alle plaatsen waarnaar een schaakstuk zich kan verplaatsen op het gegeven speelbord. Om dit onderdeel te halen, is het voldoende indien voor de andere stukken deze methode een lege lijst geeft.

```
S=Schaakbord()
S.zet_begin_positie()
p=S.stuk_op((2,1))
print(p.geldige_zetten(S))     # Output: [(3,1),(4,1)]
S.verplaats(p,(4,1))
print(p.geldige_zetten(S))     # Output: [(5,1)]
S.verplaats(p,(5,1))
print(p.geldige_zetten(S))     # Output: [(6,1)]
S.verplaats(p,(6,1))
print(p.geldige_zetten(S))     # Output: [(7,2)]
S.verplaats(p,(7,2))
print(p.geldige_zetten(S))     # Output: [(8,1),(8,3)]
```

- Voeg een methode `Speel(Schaakstuk,(rij,kolom))` toe aan de klasse "Schaakbord". Deze methode verplaatst het schaakstuk naar de positie (rij,kolom) indien dit toegestaan is. Indien de verplaatsing niet is toegestaan, wordt een foutboodschap afgedrukt.

```
S=Schaakbord()
S.zet_begin_positie()
p=S.stuk_op((2,1))
S.speel(p,(5,2))        # Output: "Ongeldige zet: Witte pion kan enkel
                        #        verplaatst worden naar: [(3,1),(4,1)]"
S.speel(p,(4,1))        # Geen output
print(S.stuk_op((2,1))) # Output: None
```

Iedereen speelt mee

Breid de functie `geldige_zetten(Schaakbord)` uit naar de andere schaakstukken. Je moet de methode `geldige_zetten(Schaakbord)` dus implementeren voor de volgende klassen:

- Toren
- Paard
- Loper
- Koning
- Koningin

Pas indien nodig de methode `speel(Schaakstuk,(rij,kolom))` aan in de klasse "Speelbord" om de bewegingen van de andere stukken mogelijk te maken. Opnieuw hoef je geen rekening te houden met "Schaak" en "Schaakmat".

Check ... and Mate

Een essentiëel onderdeel van het schaakspel, je raadt het nooit, zijn de begrippen "schaak" en "schaakmat".

- Voeg aan de klasse Schaakbord een methode `schaak(kleur)` toe, die aangeeft of de koning van de gegeven kleur al dan niet schaak staat:

```
S=Schaakbord()
pw=Pion(WIT)
kz=Koning(ZWART)
pz=Pion(ZWART)
kw=Koning(WIT)

S.plaats(pw,(5,3))      #  .  .  .
S.plaats(pz,(5,5))      #  .  Kz .
S.plaats(kz,(6,4))      #  Pw .  Pz
S.plaats(kw,(4,3))      #  Kw .  .

print(S.schaak(ZWART))  #  Output: True
print(S.schaak(WIT))    #  Output: False

# wordt vervolgd ...
```

- Pas de methode `Speel(Schaakstuk,(rij,kolom))` van de klasse Schaakbord en/of de methodes `geldige_zetten(Speelbord)` van de schaakstukken aan zodat zetten die ervoor zorgen dat de eigen koning schaak komt te staan, niet toegestaan zijn.

```
# ... vervolg

S.speel(Kz,(5,3))       #  Output: "Ongeldige zet:
                        #  Zwarte koning kan enkel
                        #  verplaatst worden naar:
                        #  [(6,3),(7,3),(7,4),(7,5),(6,5)]"
```

- Implementeer een methode `schaakmat(kleur)` die aangeeft of de koning van de aangegeven kleur al dan niet schaakmat staat:

```
S=Schaakbord()
pw1=Pion(WIT)
kz=Koning(ZWART)
pw2=Pion(WIT)
kw=Koning(WIT)

S.plaats(kz,(8,8))      #  f  g  h
S.plaats(pw1,(7,7))     #  .  .  Kz    8
```

```
S.plaats(pw2,(7,6))     #  Pw Pw .     7
S.plaats(kw,(6,8))      #  .  .  Kw    6

print(S.schaakmat(ZWART))  #  Output: True
```

- Implementeer een methode `pat(kleur)` die aangeeft of de koning van de aangegeven kleur al dan niet pat staat. Pat wil zeggen dat de koning niet schaak staat, maar dat er geen enkele geldige beweging van eender welk stuk van die kleur mogelijk is:

```
S=Schaakbord()
pw1=Pion(WIT)
kz=Koning(ZWART)
kw=Koning(WIT)

                        #  f  g  h
S.plaats(kz,(8,8))      #  .  .  Kz    8
S.plaats(pw1,(7,6))     #  Pw .  .     7
S.plaats(kw,(6,8))      #  .  .  Kw    6

print(S.pat(ZWART))     #  Output: True
```

Let's Play!

Nu alle stukken kunnen bewegen en we schaak en schaakmat kunnen detecteren, is de tijd rijp om alles te integreren. Maak een interface voor jouw programma om met twee spelers te schaken. Hieronder vind je een mogelijke sequentie van interacties met het programma. De tekst na de dubbele punt in de listing wordt door de spelers ingegeven:

```
Welkom! Een nieuw spel wordt gestart.

    a  b  c  d  e  f  g  h

8   Rb Nb Bb Qb Kb Bb Nb Rb
7   Pb Pb Pb Pb Pb Pb Pb Pb
6   .  .  .  .  .  .  .  .
5   .  .  .  .  .  .  .  .
4   .  .  .  .  .  .  .  .
3   .  .  .  .  .  .  .  .
2   Pw Pw Pw Pw Pw Pw Pw Pw
1   Rw Nw Bw Qw Kw Bw Nw Rw

Wit aan zet: e2-e4
```

```
    a  b  c  d  e  f  g  h

8   Rb Nb Bb Qb Kb Bb Nb Rb
7   Pb Pb Pb Pb Pb Pb Pb Pb
6   .  .  .  .  .  .  .  .
5   .  .  .  .  .  .  .  .
4   .  .  .  Pw .  .  .  .
3   .  .  .  .  .  .  .  .
2   Pw Pw Pw .  Pw Pw Pw Pw
1   Rw Nw Bw Qw Kw Bw Nw Rw

Zwart aan zet: e7-e5


    a  b  c  d  e  f  g  h

8   Rb Nb Bb Qb Kb Bb Nb Rb
7   Pb Pb Pb .  Pb Pb Pb Pb
6   .  .  .  .  .  .  .  .
5   .  .  .  Pb .  .  .  .
4   .  .  .  Pw .  .  .  .
3   .  .  .  .  .  .  .  .
2   Pw Pw Pw .  Pw Pw Pw Pw
1   Rw Nw Bw Qw Kw Bw Nw Rw

Wit aan zet: h7-h5
Ongeldige zet! (geen wit stuk op positie h7)
Wit aan zet: b1-c3


    a  b  c  d  e  f  g  h

8   Rb Nb Bb Qb Kb Bb Nb Rb
7   Pb Pb Pb .  Pb Pb Pb Pb
6   .  .  .  .  .  .  .  .
5   .  .  .  Pb .  .  .  .
4   .  .  .  Pw .  .  .  .
3   .  .  Nw .  .  .  .  .
2   Pw Pw Pw .  Pw Pw Pw Pw
1   Rw .  Bw Qw Kw Bw Nw Rw
```

Het spel moet schaak, pat en schaakmat kunnen detecteren. Het spel stopt bij pat en schaakmat.

Praktisch

Stuur je oplossing voor de deadline in op Blackboard, in de vorm van een ZIP-bestand met daarin alle benodigde bestanden. Inzendingen via e-mail of na het verlopen van de deadline zullen niet worden geaccepteerd. Je wordt verwacht om individueel aan deze opdracht te werken, zonder plagiaat te plegen. Noem je file schaken.py en test deze door middel van Schaken_Test_studenten.py. Zorg ervoor dat je code met deze file getest kan worden, zonder te crashen; inzendingen die hiertoe niet in staat zijn kunnen geen voldoende op deze opdracht halen. Deze taak staat op 12 punten:

- 10 punten op correctheid van je code (2 punten per onderdeel)
- 0.5 punt op het volgen van PEP8 conventies
- 0.5 punt op het schrijven van *zinvolle* commentaar
- 0.5 punt op efficintie (zowel qua tijd als geheugen)
- 0.5 punt op stijl (o.a. duplicate code)