Uitwerking opdracht

Opdracht :	*Paardensprong*
Weeknummer :	*1*
Studentnummer :	*s1060679*
Naam student :	*Dion van den Berg*
Specialisatie :	*MEDT*
Pogingnummer :	*1*

1. Vraagstelling

De gebruiker wil weten of de zet die hij wil maken met zijn paard ook daadwerkelijk kan.

2. Specificatie

Invoer

De gebruiker voert zijn begin positie in en zijn eind positie met een letter en een cijfer.

Uitvoer

De gebruiker krijgt een Ja of NEE op de vraag of hij de stap kan maken

Verband tussen in- en uitvoer

De uitvoer is een berekening of er een L vorm is gemaakt van de stap.

Beperkingen

Je kunt geen halve nummers invoeren
Je moet er van uitgaan dat de gebruiker niet meer dan de letter H en het cijfer 6 invoert

Voorbeelden (testscenario's)

Test 1
Invoer
Positie 1: A1
Positie 2: B3

Uitvoer:
Deze sprong kan gemaakt worden van A1 naar B3

Test 2
Invoer
Positie 1: C4
Positie 2: C5

Uitvoer:
Deze sprong kan niet gemaakt worden van A1 naar B3

3. Ontwerp

Hoe vang ik de invoer op van de gebruiker ?
Door de python functie input

Hoe ga ik rekenen met een letter ?
Door het om te zetten naar de ascii decimaal waarde.

Hoe laat ik alles uiteindelijk zien ?
in 1 regel tekst wordt de input gegeven van de gebruiker en de uitkomst

4. Pseudocode

```
Invoer1 = Invoer gebruiker positie 1
Invoer2 = Invoer gebruiker positie 2

Functie omzetten( string )
        X = int van invoer1[1]
        Y = int van de ascii waarde van invoer[0] + 1

Functie check ( int, int ) -> boolean
        Invoer1[0] – invoer2[0] == 2 en invoer1[1] – invoer2[1] == 1
        Of
        Invoer1[0] – invoer2[0] == 1 en invoer1[1] – invoer2[1] == 2

Main()
Print “kan wel of niet door Check (omzetten(invoer1), omzetten(invoer2)”
```

5. Code

```
def coordinates(positie: str)->tuple:

    # positie komt binnen als een positie op het schaakbord als een
letter en een cijfer bijv. D6
    # het cijfer hoeft niets meer mee te gebeuren.
    x = int(positie[1])

    # van de letter maak ik ook een cijfer vanuit de ascii tabel
    y = ord(positie[0]) + 1

    # return dit als een tuple om er later los mee te kunnen rekenen.
    return x, y


def checker(begin: tuple, eind: tuple)->bool:
    # return een boolean of de L vorm gemaakt kan worden of niet.
Richting maakt daarvoor niet uit.
    return (abs(begin[0] - eind[0]) == 2 and abs(begin[1] - eind[1]) ==
1) or (abs(begin[0] - eind[0]) == 1 and abs(begin[1] - eind[1]) == 2)

# print een intro met uitleg
print("Voer eerst de beginpositie in en daarna de eindpositie van het
schaakstuk.\nDeze invoer moet wel op het veld bestaan.\nGebruik dus A
t/m H en 1 t/m 6")

# vraag de gebruiker om de posities
beginPos = input("De startpositie: ")
eindPos = input("De eindpositie: ")

# zet de posities om in ints om later te gebruiken in de methode
checker.
begin = coordinates(beginPos.lower())
eind = coordinates(eindPos.lower())

#print de uitslag. Hierin wordt de ckeck functie aangeroepen om te
kijken of de beweging wel valide is of niet.
print("De sprong kan {0}gemaakt worden van {1} naar {2}".format(["niet
", ""][checker(begin, eind)], beginPos, eindPos))
```

6. Test

Test 1

Voer eerst de beginpositie in en daarna de eindpositie van het schaakstuk.
Deze invoer moet wel op het veld bestaan.
Gebruik dus A t/m H en 1 t/m 6
De startpositie: C4
De eindpositie: C5
De sprong kan niet gemaakt worden van C4 naar C5

Test 2

Voer eerst de beginpositie in en daarna de eindpositie van het schaakstuk.
Deze invoer moet wel op het veld bestaan.
Gebruik dus A t/m H en 1 t/m 6
De startpositie: A1
De eindpositie: B3
De sprong kan gemaakt worden van A1 naar B3