Gmail **Stijn C <creemers.stijn13@gmail.com>**

Introductieproject Blackjack

3 messages

Stijn C <creemers.stijn13@gmail.com> Sat, Mar 22, 2025 at 12:54 PM
To: jef@koto.build

Hoi Jef,

In bijlage kan je mijn Blackjack-project terugvinden. Om het nog even te kaderen: het grote merendeel van de code werd opgemaakt aan de hand van een tutorial die te volgen was op Youtube.
Ik heb hierbij vooral geprobeerd om alles mee handmatig over te nemen en niet gewoon een copy-paste te doen vanuit de github van degene die de tutorial presenteerde. Kwestie van mijn muscle memory al wat te trainen.
Alle termen, functies, variables etc die ik niet kende heb ik dan opgezocht en kort beschreven in een apart bestand voor mezelf, zodat het ook duidelijk werd wat ik juist aan het overnemen was.

Daarna heb ik nog wat nagedacht over simpele toevoegingen die ik zelf kon doen. Als eerste heb ik 6 kleine diamantjes toegevoegd rond de win-boodschap om het wat vrolijker te maken.
Daarbij heb ik ook een wisselende achtergrondkleur gemaakt waarbij er bij verlies een rode kleur komt en bij winst een groene kleur.

Ik weet niet of er nog echt bugs in gaan zitten aangezien ik het programma zeer regelmatig eens heb getest om te kijken of alles nog klopte en Pycharm al een grote hulp was om de aanwezige fouten eruit te halen.
Laat me vooral weten als er iets is wat ik zou kunnen aanpassen of toevoegen. Alle opmerkingen zijn welkom!

Groetjes
Stijn

blackjackdraft - kopie.txt
10K

Jef Stals <Jef@koto.build> Sun, Mar 23, 2025 at 8:16 AM
To: Stijn C <creemers.stijn13@gmail.com>

Hey Stijn,

Wow, mooi project !
Ik heb het spel gespeeld, en dat lijkt behoorlijk goed te werken. Ik denk voor een eerste python project ziet er dat eigenlijk goed uit.

Als ik dan wat feedback moet geven:

- Bij ‘deal_cards’:

```
def deal_cards(current_hand, current_deck):
    card = random.randint(0, len(current_deck))
    current_hand.append(current_deck[card-1])
    current_deck.pop(card-1)
    return current_hand, current_deck
```

- Ik zou ‘card’ hier ‘card_index’ noemen, omdat het over de index van de current_deck gaat. Zo weet je dat het over een ‘getal gaat’.
- En ik zou dan de ‘-1’ logica toepassen op de eerste lijn, en dan vervolgens de index gebruiken. Dit maakt het iets eenvoudiger dan telkens ‘-1’ te moeten toevoegen in elke regel. In deze functie is het natuurlijk duidelijk en overzichtelijk, maar in complexere logica kan die ‘-1’ verwarrend zijn.
    - Als je het zo schrijft, dan ga je wss ook snel zien dat ‘card -1’ theoretisch -1 zou kunnen worden. Wat geen geldige index is, en dus volgens mij een kleine bug is? De kans dat die bug voor komt is 1 / 208, dus moeilijk eruit te halen.
    - Je kan ook eens kijken of je niet ‘random.randrange’ kan gebruiken.
    - Andere oplossing is: card_index = random.randint(0, len(current_deck)-1)
- Je zou ook heel de logica van random kaarten ook kunnen weg laten, en gewoon in het begin je deck ‘shuffelen’. (https://www.w3schools.com/python/ref_random_shuffle.asp)

- Kaarten tekenen:
    - Ik zou hier een aparte functie van maken en paar constante gebruiken, zodat alle logica van de kaart tekenen op één plaats zit.
    - VB:
- `pygame.draw.rect(screen, 'white', [70 + (70 * i), 460 + (5 * i), 120, 220], 0, 5)`
    - Die 70 is wss uw breedte ofzo, dat zou ik ergens als constante zetten, zodat als je ooit een kaart_breedte wil aanpassen je niet overal in je code moet zoeken.
    - Het ding is: nu is dat wss duidelijk waarom die code zo opgebouwd is, maar als je dat over een half jaar opnieuw bekijkt moet je wss beginnen puzzelen. Hier dus ook best aan de hand van variabelen / constante en functies duidelijk laten zien wat je doet, zodat je later nog aan je code uit kunt.

- De meningen over comments in code zijn verdeeld, maar in sommige stappen zou ik misschien nog wat comments toevoegen om de code vlugger leesbaar te maken.
  Bijvoorbeeld:

  one_deck = 4 * cards → Hier moest ik even nadenken waarom dat * 4 is (uiteraard logisch, maar met een korte comment is het direct duidelijk)

- Naamgeving variabelen: de meningen zijn hier ook over verdeeld, en afhankelijk met wie je praat ga je een andere suggestie horen. Dus ik denk dat je moet kijken wat voor jezelf werkt. Maar ik verkies altijd variabelen die duidelijk omschrijven waar het over gaat. Bijvoorbeeld:

```
def draw_cards(player, dealer, reveal):
```

player → player_cards , want dan weet je direct dat het over kaarten gaat. Nu zou het ook nog over de player zelf kunnen gaan.

- Leesbaarheid van code: (opnieuw, meningen zijn hier over verdeeld)
  - Basis principe is dat een functie / blok atlijd in één schermhoogte past (dat is goed gedaan)
  - Ander principe is dat je via kleinere functies eigenlijk duidelijker maakt wat de code doet.
    - Op meerdere plaatsen komt zoiets voor:

```
active = True
        initial_deal = True
        game_deck = copy.deepcopy(decks * one_deck)
        my_hand = []
        dealer_hand = []
        outcome = 0
        hand_active = True
        reveal_dealer = False
        outcome = 0
        add_score = True
```

Dit zou je in een aparte functie kunnen zetten dat omschrijft wat het doet. Vb. ‘start_new_game’. Op deze manier geeft de naam van de functie weer wat het doet, en dient het ook een stukje als comment. Daarnaast wordt je for-, if- structuur compacter en makkelijker leesbaar.

- Er zijn ook suggesties rond hoe diep je een if-else structuren mag nesten.
- En meer geavanceerd: je kan meestal code ook herschrijven naar een structuur dat je nooit een ‘else’ gebruikt door slim gebruik te maken van een ‘return’. Da zou de leesbaarheid van de code ook ver hogen.

- Waarschijnlijk niet in scope van het project, maar Object Oriented Programing (OOP) zou hier wel een meerwaarde zijn. Je krijgt dan verschillende entiteiten (Card, Deck, Player, Dealer, …). De structuur wordt dan nog duidelijker, en het maakt dat je stukken kunt herbruiken en het makkelijker schaalbaar wordt.

Dus globaal gezien, mooi werk :--)

Laat maar weten als ergens verheldering bij mijn feedback nodig is.

Met vriendelijke groet,

Jef

Jef Stals

Founder, Koto

Parkdreef 12, 3960 Bree, België

+32 478 37 12 16

www.koto.build

[Quoted text hidden]

Stijn C <creemers.stijn13@gmail.com> Sun, Mar 23, 2025 at 1:44 PM
To: Jef Stals <Jef@koto.build>

Hoi Jef,

Heel erg bedankt voor de uitgebreide feedback, daarmee kan ik zeker wat aanpassingen doen.
Merci voor al de moeite die je erin hebt gestoken!

Groetjes,
Stijn
[Quoted text hidden]