Fold, Check, Call, Raise...

Poker je populárna kartová hra a z jeho veľa variantov je najrozšírenejší Texas hold 'em. Cieľom v pokri je vykombinovať čo najsilnejšiu päticu kariet podľa definovaných výherných kombinácií. V hold 'eme každý hráč dostane dve karty a následne na stôl je postupne položených päť kariet, ktoré sú rovnaké pre každého hráča. Víťaz sa určí tak, že každý hráč si vykombinuje najlepšiu kombináciu pätice kariet z dostupných sedem (päť na stole, dve v ruke) a tieto kombinácie sa porovnajú.

V Texas hold 'eme pritom sú uznávané nasledovné kombinácie (zostupne od najsilnejšieho):

1. Royal Flush – postupka kariet 10 J Q K A v tej istej farbe
2. Straight Flush – akákoľvek postupka piatich kariet rovnakej farby
3. Four of a Kind – štvorica kariet s rovnakou hodnotou
4. Full House – trojica a dvojica kariet s rovnakou hodnotou
5. Flush – päť kariet tej istej farby
6. Straight – postupka kariet rôznych farieb
7. Three of a Kind – trojica kariet s rovnakou hodnotou
8. Two Pairs – dve dvojice kariet s rovnakou hodnotou
9. One Pair – dvojica kariet s rovnakou hodnotou
10. High Card – ak ani jeden hráč nemá kombinácie, vyhrá ten s kartou s najvyššou hodnotou (karty sú zoradené 2 – A)

Ak teda máte Royal Flush, s istotou vyhráte. Ak dvaja hráči majú kombináciu na rovnakej úrovni, tiež sa rozhodne štýlom High Card. Napríklad ak máme dvoch hráčov s dvomi pármi 10 a 9, a jeden má navyše Q, kým druhý 8, tak vyhrá prvý (má vyššiu kartu). Keby okrem párov by obaja mali kartu s rovnakou hodnotou, nastane remíza. Podrobnejší popis pravidiel nájdete na wikipédii: https://en.wikipedia.org/wiki/Texas_hold_%27em#Examples.

Na stôl sú položené najprv tri karty (*flop*), potom jedna (*turn*) a ešte jedna (*river*), pričom po každom kole rozdávania nasleduje stávkové kolo, v ktorom hráči majú k dispozícii štyri akcie:

- položiť karty (*fold*) – hráč skladá karty a preňho skončí kolo;
- pokračovať (*check*) – hráč nerobí nič a nasleduje ďalší hráč;
- dorovnať (*call*) – hráč dorovná stávku;
- prihodiť (*raise*) – hráč navyšuje stávku.

Na konci stávkového kola musí každí hráč dorovnať stávku alebo skladať karty. Keďže ale škola nesmie podporovať hazardné hry, v tomto zadaní nás bude zaujímať poker z hľadiska pravdepodobnosti: naším cieľom je vypočítať pravdepodobnosť výhry jednotlivých hráčov.

Pri hracích kartách existuje ale obrovské množstvo možností. Balíček 52 kariet napríklad môžete zamiešať 52! spôsobmi (FYI, je to číslo: 80 658 175 170 943 878 571 660 636 856 403 766 975 289 505 440 883 277 824 000 000 000 000), a práve preto nie je reálne aby sme nasimulovali všetky možné kolá hociktorej kartovej hry – keby ste mali bilión počítačov, každý bilión procesorov, a každý by dokázal nasimulovať tisíc unikátnych zoradení bilión balíčkov kariet za

sekundu, a robili by to od vzniku vesmíru, tak prvé duplikáty by sa objavili až teraz. To bohužiaľ znamená, že nedokážete nájsť výhernú stratégiu pre každý prípad.

Ale späť k nášmu zadaniu: pre jednoduchosť a pre vyriešiteľnosť úlohy budeme považovať iba jeden jediný okamih v pokri, a to moment pred riverom s dvomi hráčmi. Na stole teda už máme štyri karty a poznáme karty, ktoré držia v ruke hráči, a takto nám ostáva 44 možných výsledkov. Cieľom bude určiť pravdepodobnosť toho, že vyhrá hráč 1, hráč 2, alebo nastane remíza (v pokri dosť nepravdepodobné).

Kostru riešenia nájdete v súboru `problem2.py`. Karty budeme reprezentovať ako dvojicu hodnôt, pričom prvá hodnota je farba karty a druhá je jej hodnota. Farbu budú reprezentovať písmená H (hearts – srdcia), C (clubs – kríže), S (spades – piky) a D (diamonds – kára). Hodnoty budeme reprezentovať číslami od 2 po 14 (J – 11, Q – 12, K – 13, A - 14). Aj keď pôvodne eso môžeme považovať za kartu po kráľovi alebo pred dvojkou, my ho budeme považovať pre jednoduchosť iba za hodnotu 14, teda eso príde vždy po kráľovi.

Skript obsahuje dve pomocné funkcie. Funkcia `generate_setup` Vám vygeneruje náhodný príklad vstupu, teda zoznam dvoch kariet pre hráča 1, rovnaký zoznam pre hráča 2, trojicu kariet pre flop, a zoznam jednej karty pre turn. Takéto údaje budú vstupom pre našu hlavnú funkciu.

Druhá pomocná funkcia je evaluate_hand, ktorá vyhodnotí päticu kariet – samotné rozpoznávanie kombinácií však musíte implementovať vy. Návratová hodnota funkcie reprezentuje relatívnu silu rôznych kombinácií. Ako môžete vidieť, kontrolné funkcie sa zavolajú v zostupnom poradí, čo nám zjednoduší implementáciu funkcií pre rozpoznávanie kombinácií. Ako príklad si zoberme päticu reprezentujúcu Royal Flush. Vďaka postupnému vyhodnoteniu takúto päticu môžeme považovať za Royal Flush, Straight Flush, Flush, aj Straight, do úvahy sa berie tak či tak najvyššia kombinácia. Takto budete mať ľahšiu úlohu pri implementácii vyhodnocovacích funkcií.

Úloha 1 – 3 body

V prvom kroku implementujete krátke funkcie, ktoré vyhodnotia, či istá pätica kariet spĺňa kritéria istej kombinácie. Vstupom je vždy zoznam piatich dvojíc, ktoré reprezentujú karty. Vstup bude vždy platný, takže nemusíte riešiť prípad, že by zoznam reprezentoval nereálnu päticu kariet (duplicitné karty, karty neznámej farby alebo hodnoty) – môžete tak ale urobiť. Návratová funkcia týchto funkcií sú dve hodnoty: `boolean`, ktorý určí, či karty spĺňajú kritériá kombinácie; a `tuple` s hodnotami relevantných kariet.

Úloha 1.1 – `is_royal_flush` – 0,2 b

Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Royal Flush, v opačnom prípade `False`. Karty musia byť rovnakej farby a musia mať hodnoty 10, 11, 12, 13, 14.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota najväčšej karty, teda 14. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.2 – `is_straight_flush` – 0,4 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Straight Flush, v opačnom prípade `False`. Karty musia byť rovnakej farby a musia mať hodnoty z ľubovoľnej postupnosti. Prípad, že postupnosť začína esom (A – 2 – 3 – 4 – 5) nebudeme považovať za platný. Funkcia vracia hodnotu `True` aj pre kombinácie Royal Flush.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota najväčšej karty. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.3 – `is_four_of_a_kind` – 0,4 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Four of a Kind, v opačnom prípade `False`. Štyri karty z pätice musia mať rovnakú hodnotu.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota kariet, z ktorých sú štyri. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.4 – `is_full_house` – 0,4 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Full House, v opačnom prípade `False`. V pätici kariet tri musia mať rovnakú hodnotu a dve musia mať tiež rovnakú hodnotu (inú ale od trojice) – Full House je teda kombinácia trojice a dvojice kariet rovnakej hodnoty.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s dvomi prvkami – hodnota trojice kariet a hodnota páru. Ak teda máme tri osmičky a dve štvorky, funkcia vráti `True, (8, 4)`. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.5 – `is_flush` – 0,2 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Flush, v opačnom prípade `False`. Všetky karty v pätici musia byť rovnakej farby. Funkcia vracia hodnotu `True` aj pre kombinácie typu Royal Flush.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota najväčšej karty. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.6 – `is_straight` – 0,4 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Straight, v opačnom prípade `False`. Karty musia reprezentovať postupnosť, na ich farbe ale nezáleží. Funkcia vracia hodnotu `True` aj pre kombinácie typu Royal Flush a Straight Flush.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota najväčšej karty. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.7 – `is_three_of_a_kind` – 0,2 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Three of a Kind, v opačnom prípade `False`. Pätica kariet musí obsahovať presne tri s rovnakou hodnotou. Funkcia vracia hodnotu `False` pre kombinácie typu Four of a Kind.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota kariet, z ktorých sú tri. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.8 – `is_two_pairs` – 0,4 b
Funkcia vracia `True` ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie Two Pairs, v opačnom prípade `False`. Pätica kariet musí obsahovať dve dvojice kariet s rovnakou hodnotou, tieto dvojice ale nesmú mať rovnakú hodnotu – funkcia vracia `False` pre kombinácie typu Four of a Kind.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s dvomi prvkami – hodnota kariet v prvom a druhom páry, zostupne. Pre dva páry 7 a 11 bude `tuple (11, 7)`. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Úloha 1.9 – `is_pair` – 0,4 b
Funkcia vracia `True`, ak pätica kariet vo vstupnom parametri `hand` spĺňa podmienky kombinácie One Pair, v opačnom prípade `False`. Pätica kariet musí obsahovať presne jednu dvojicu kariet s rovnakou hodnotou. Funkcia vracia `False` pre kombinácie typu Two Pairs a Four of a Kind.
Prvá návratová hodnota je `True` alebo `False`, a druhá je `tuple` s jedným prvkom – hodnota kariet v páry. Ak je prvá návratová hodnota `False`, namiesto `tuple` funkcia vráti hodnotu `None` alebo prázdny `tuple`.

Príklady pre rôzne kombinácie a ich vyhodnotenie jednotlivými funkciami nájdete v súbore `eval.csv`.

Poznámka: Pre získanie bodov pre úlohy 1.1 – 1.9 potrebujete implementovať funkcie, nestačí zapísať defaultné návratové hodnoty.

Úloha 2 – 1 bod
V úlohe 2 naprogramujeme funkcie, ktoré nám pomôžu vybrať výhernú kombináciu z dvoch možností. Vstupy reprezentujú dve pätice kariet – päticu od hráča 1 a päticu od hráča 2. Môžete pritom použiť pomocnú funkciu `evaluate_hand`.

Úloha 2.1 – `compare_highest_card` – 0,5 b
Funkcia vracia výhernú päticu kariet z dvoch (vstupné parametre `hand_1` a `hand_2`). Kým funkcia `evaluate_hand` nám pomôže pri výbere výhernej kombinácie ak hráči majú kombinácie na rôznej úrovni, nevie ale rozpoznať výhernú kombináciu ak obaja majú napríklad trojicu rovnakých kariet. Funkcia `compare_highest_card` rieši presne tento prípad, a vyberie výhernú päticu na základe karty s najvyššou hodnotou. Ak sa nedá rozhodnúť medzi dvomi päticami – všetky karty majú rovnakú hodnotu – funkcia vracia hodnotu `None`.

Pomôcka: Najprv by ste mali identifikovať jedinečné karty v oboch päticiach (zbytočne by ste porovnávali tie isté karty) a potom nájsť karty s najvyššou hodnotou. Ak je to karta s rovnakou hodnotou ale inej farby pre obe pätice, rozhodne druhá najvyššia karta, potom tretia, atď.

Úloha 2.2 – `find_better` – 0,5 b

Funkcia `find_better` nájde výhernú päticu z dvoch pätíc a vracia ju (teda návratová hodnota je jeden z vstupných parametrov `hand_1` alebo `hand_2`). V prípade remízy funkcia vracia `None`.

Ak nastane rovnosť kombinácií, najprv skontrolujte hodnoty relevantných kariet (druhá návratová hodnota z evaluácie). Prednostne vyhrá hráč, ktorý má vyššie relevantné karty (napr. vyšší pár, vyššiu trojicu, vyššiu postupku, atď.), ak obaja majú kombináciu na rovnakej úrovni, do úvahy sa berie najvyššia karta. Ak aj najvyššie karty sú rovnaké, nastane remíza a funkcia vracia hodnotu `None`.

Vo funkcii použite `evaluate_hand`, aby ste zistili, ktorý hráč akú kombináciu má, a `compare_highest_card` pre prípad, že obaja majú kombináciu na rovnakej úrovni.

V súbore `pick.csv` nájdete ukážkové príklady s ich správnymi výstupmi pre funkciu `find_better`.

Úloha 3 – 1 bod

Po predošlej úlohe už dokážeme vybrať výhernú päticu z dvoch. Avšak v Texas hold 'eme hráč si môže zostrojiť päticu zo siedmich kariet (päť kariet na stole, dve v ruke). V tomto kroku riešenia pomôžeme hráčom nájsť tú najsilnejšiu päticu z dostupných kariet.

Úloha 3.1 – `get_all_combinations` – 0,5 b

Najprv ale musíme získať všetky možnosti výberu kariet – existuje 21 rôznych spôsobov vybrať päť kariet zo siedmich. Na poradí kariet pritom nezáleží: päticu C7, H6, D2, S3, C8 a päticu C7, D2, S3, C8, H6 budeme považovať za tú istú (karty sú iba pomiešané). Pozor, hráč svoje karty nemusí použiť!

Implementujte funkciu `get_all_combinations`, ktorá zostrojí a vracia zoznam všetkých možných kombinácií výberu piatich kariet zo siedmich. Funkcia má štyri vstupné parametre, pričom všetky sú zoznamy dvojíc, kde dvojice reprezentujú jednotlivé karty:

- `hand` – dvojica kariet v ruke hráča
- `flop` – trojica kariet na stole
- `turn` – jedna karta na stole
- `river` – posledná karta na stole.

Pomôcka: V štandardných moduloch jazyka Python nájdete funkciu, ktorá slúži presne na vygenerovanie kombinácií výberu niekoľkých elementov z populácie. Potrebujete pritom iba opraviť vstupy a výstupy.

Úloha 3.2 – `select_best` – 0,5 b
Ak už vieme vygenerovať všetky možné kombinácie pätíc, vieme určiť aj tú najlepšiu z nich, v čom nám pomôže funkcia `select_best`. Táto funkcia má rovnaké vstupné parametre ako `get_all_combinations`, vracia ale iba jeden zoznam piatich kariet (karty sú reprezentované ako dvojica hodnôt), a to najsilnejšiu päticu kariet.

V súbore `select.csv` nájdete ukážkové príklady s ich správnymi výstupmi pre funkciu `select_best`.

Úloha 4 – 1 bod
Všetko je už pripravené pre implementáciu hlavnej funkcie, ktorá odpovie na otázku, ktorú riešime v tomto zadaní: aká je pravdepodobnosť výhry jednotlivých hráčov, resp. remízy?

Implementujte funkciu `calculate_chances`, ktorá nasimuluje všetky možné výsledky hry na základe vstupných hodnôt a vracia tri hodnoty, a to pravdepodobnosť výhry hráča 1, pravdepodobnosť výhry hráča 2, pravdepodobnosť remízy. Pravdepodobnosť vypočítajte ako podiel počtu možností pre daný výsledok a počtu všetkých možných výsledkov – bude to hodnota v rozmedzí 0 až 1.

Funkcia má štyri vstupné parametre:
- `player1_hand` – zoznam dvoch kariet, ktoré drží v ruke hráč 1
- `player2_hand` – zoznam dvoch kariet, ktoré drží v ruke hráč 2
- `flop` – zoznam troch kariet položených na stôl v prvom kole
- `turn` – zoznam jednej karty, ktorá bola položená na stôl ako posledná

Pomôcka: Pri riešení úlohy potrebujete nasimulovať všetky možné výsledky podľa karty v river, t.j. kto by vyhral, keby sme na stôl položili túto kartu?

Ukážkové príklady pre vstup a výstup nájdete v súbore `chances.csv`.

Kostra riešenia obsahuje ešte hlavnú funkciu `main`, ktorú môžete využiť na testovanie. Pri riešení môžete vytvoriť ľubovoľné pomocné funkcie a môžete použiť hotové riešenia z ľubovoľného štandardného modulu jazyka Python.

Vaše riešenia môžete otestovať aj pomocou sady testov v súbore `problem2_sample_tests.py`. K spusteniu testov potrebujete mať nainštalovanú knižnicu `pandas`. Pri hodnotení vášho riešenia použijeme podobné testy, avšak bude ich viac.

Ak chcete otestovať nové kombinácie rozdelenia kariet, môžete použiť online nástroje ako napríklad https://www.pokerlistings.com/online-poker-odds-calculator.