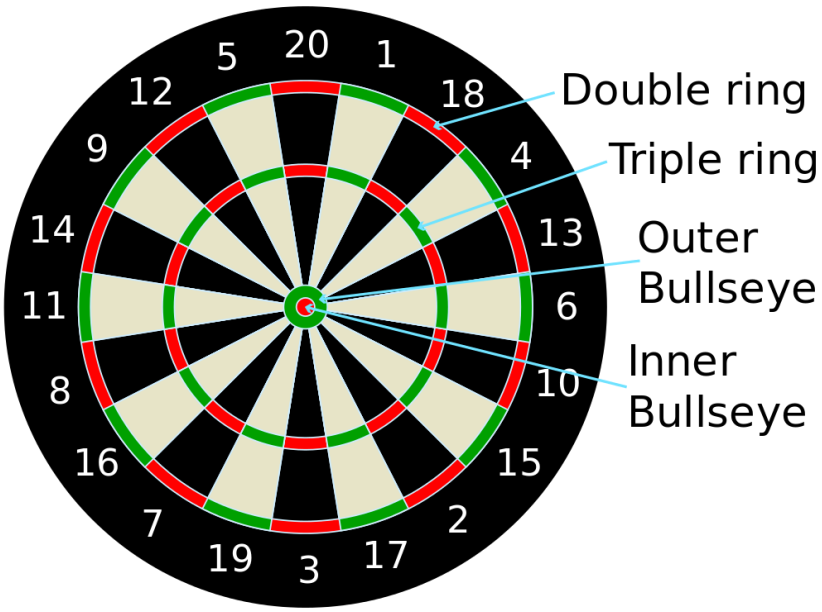
Šípky

1. zadanie – deadline: 15. 3. 2024

Šípky sú hra známa z barov a iných podnikov, v posledných rokoch sa však ich popularita rastie a niektoré turnaje sa stali ostro sledovanými prenosmi športových kanálov. Hru hrajú dvaja hráči, ktorí striedavo hádžu (obaja v jednom kole) tromi šípkami na terč, ktorý je rozdelený na niekoľko polí. Na základe toho, kde hráč trafí (alebo netrafí) terč, sa odpočítavajú body a hru vyhrá ten, ktorý najskôr dosiahne 0 bodov. Výhra je označená ako leg.

Terč je rozdelený na 20 rovnako široké segmenty, ktorým prislúcha rôzny počet bodov, ktorý sa odrátava z aktuálneho počtu bodov hráča. Ako môžete vidieť na obrázku nižšie, na terči sú označené aj dva ďalšie kruhy. Ak šípka padne na vnútorný kruh (triple ring), znamená to, že hráčovi sa odčítava trojnásobok počtu bodov daného segmentu. Podobne, ak šípka padne na vonkajší kruh (double ring), hráčovi sa odčítava dvojnásobok počtu bodov daného segmentu. Okrem toho v strede terču vidíte zelený (outer bullseye) a v ňom červený kruh (inner bullseye). Outer bullseye má hodnotu 25 bodov, inner bullseye skóruje 50 bodov. Maximálny dosiahnuteľný počet bodov v rámci jedného kola pre hráča je 180 (trikrát triple 20). Samozrejme ak hráč netrafí terč, nezíska žiadne body, v kole však môže pokračovať.



Zdroj: https://en.wikipedia.org/wiki/Darts#/media/File:Dartboard_diagram.svg

Súťažný zápas pozostáva z niekoľkých setov, pričom každý set trvá dovtedy, kým niektorý z hráčov nevyhrá tri legy. Hráči každý leg začínajú s 501 bodmi. Na to, aby hráč vyhral celý zápas, potrebuje vyhrať niekoľko setov, pričom ich počet závisí od turnaja respektíve fázy turnaja.

Keďže hráč, ktorý hodí ako prvý má miernu výhodu, hráči otvárajú legy aj sety striedavo. To znamená, že ak napríklad hráč 1 (H1) otvorí prvý leg zápasu, druhý leg otvára hráč 2 (H2), ďalší leg zase H1, a tak ďalej. Bez ohľadu na to, kto otvára posledný leg v prvom sete, prvý leg druhého setu otvorí H2 (keďže prvý set otvoril H1). Toto striedanie vieme jednoducho pochopiť tak, že rovnaký hráč otvára všetky párne resp. nepárne sety, a rovnaké pravidlo platí pre legy v rámci setu (mení sa iba identita hráčov).

Špecifickou vlastnosťou súťažných zápasov je to, že hráč vyhrá leg iba ak svojou poslednou šípkou hodí double, teda trafí terč vo vonkajšom kruhu. To znamená, že pred poslednou šípkou musí mať párny počet bodov. Keby náhodou trafil terč tak, že by mu ostal iba jeden bod, tento hod sa ráta ako neplatný, a príde o ďalšie možnosti v danom kole. Napríklad ak hráč má 7 bodov a hodí double 3, v danom kole už končí, lebo ostal by mu jeden bod, ktorým už nemôže uzavrieť, nasleduje teda druhý hráč. Avšak ak pri 7 bodoch hodí double 2, ostávajú mu 3 body, a môže ďalej pokračovať: konkrétne musí hodiť najprv 1, aby mu ostali 2 body, a následne môže uzavrieť už len double 1. Kolo hráč ukončí predčasne aj v prípade, ak hodí viac, ako má aktuálne bodov, napríklad pri počte bodov 7 ak trafí double 4, nemôže hodiť znova (až v ďalšom kole).

Vašou úlohou je spracovať záznamy zo simulovaných zápasov šípok a vypočítať rôzne štatistiky hráčov.

Poznámka: Uvedené funkcie viete implementovať nezávisle, avšak pre úplné pochopenie úlohy odporúčame riešiť úlohy v uvedenom poradí.

Úloha 1: `throw_to_points` (0,5 bodov)

Implementuje funkciu `throw_to_points`, ktorá dostane ako parameter `throw` popis jedného hodu hráča, a vráti jedno celé číslo – počet bodov, ktoré hráč hodom získal. Vstupný parameter bude typu `string`, pričom bude to zodpovedať číslu segmentu, ktorý hráč trafil. Double bude reprezentovaný písmenom D, triple písmenom T, napr.: `'20'` → 20 bodov, `'D4'` → 8 bodov, `'T7'` → 21 bodov. Outer bullseye je reprezenzovaný reťazcom `'25'`, inner bullseye reťazcom `'50'`. Ak hráč terč netrafil, vstupný parameter má hodnotu `None` (a hráč neskóruje, očakáva sa výstup 0).

Úloha 2: `parse_throws` (2 body)

Funkcia `parse_throws` dostane ako vstupný parameter zoznam `string`ov, pričom každý z nich reprezentuje hod niektorého hráča (formát rovnaký ako v predošlej úlohe). Úlohou funkcie je spracovať tento zoznam a zistiť, ktorý hod patrí ktorému hráčovi. Výstupná hodnota funkcie je zoznam dvojíc (`tuple` s dvomi hodnotami), pričom dvojica reprezentuje hody z jedného kola. V rámci tejto dvojice máte dve n-tice (`tuple`), ktoré obsahujú hody jednotlivých hráčov. Teda každá z týchto n-tíc má maximálne tri hodnoty (môže ich však mať menej ak hráč mal neplatní hod alebo vyhral leg).

Napríklad pre jeden leg:

```
['T20', 'T20', 'T20', 'T20', 'T20', '5', 'T20', 'T20', 'T20',
'1', 'T20', 'T20', 'T20', '19', 'T20', '1', '20', '20', '1',
'D1'] → [(('T20', 'T20', 'T20'), ('T20', 'T20', '5')), (('T20',
'T20', 'T20'), ('1', 'T20', 'T20')), (('T20', '19', 'T20'),
('1', '20', '20')), (('1', 'D1'),)]
```

Pri implementácii udržiavajte aktuálny počet bodov jednotlivých hráčov (začínajú s 501), a postupne spracujte hodnoty zo zoznamu a aktualizujte tieto počty bodov. Zatiaľ sa nepotrebujete zaoberať identitou jednotlivých hráčov a ich striedaním, n-tice vytvárajte len na základe toho, či hody prislúchajú prvému alebo druhému hráčovi v rámci legu. Môžete rátať s tým, že vstupný zoznam bude vždy platný a bude obsahovať realistické údaje zo simulovaného zápasu.

Poznámka: Funkcia musí správne identifikovať aj neplatné hody. Tie zapíšete do výsledku, ale naznačujú, že kolo pre daného hráča končilo a nasleduje druhý hráč. Takto musíte brať do úvahy prípady, ak hod má vyššiu hodnotu ako je aktuálny počet bodov hráča; ak hodil hodnotu, ktorá by znamenala, že mu ostane iba jeden bod; alebo ak dostal sa na 0, ale nie doublom.

Úloha 3: `save_throws` (1 bod)

Implementujte funkciu `save_throws`, ktorá zapíše hody zo zápasu do textového súboru v štruktúrovanom formáte. Funkcia má dva vstupné parametre:

- `throw_pairs` – zoznam n-tíc s formátom ako výstup z funkcie `parse_throws()`;
- `path` – `string` reprezentujúci cestu k súboru, ktorý sa má vytvoriť.

Každý riadok v súbore reprezentuje hody oboch hráčov, pričom tieto sú oddelené medzerou. Súbor na konci musí mať presne jeden prázdny riadok. Ak hráč netrafil terč, označte to `X`, ak hráč 1 nehodil trikrát pretože bol jeho predošlý neplatný, označte to medzerou. Ak v danom kole sa hráč 2 nedostal k slovu, jeho hody vôbec nezapíšte do súboru, teda po hode hráča 1 nenasleduje ani medzera. Ak však hráč 2 mal jeden neplatný hod a tým pádom ďalšie dve šípky už nehodil, v daných riadkoch po hode hráča 1 medzera musí byť.

Napríklad pre leg:

```
[(('T20', 'T20', 'T20'), ('T20', '20', 'T20')), (('5', 'T20',
'T20'), ('1', '20', '1')), (('T20', '1', 'T20'), ('T20', 'T20',
'T20')), (('T19', 'D14'), ('T1', '5', 'T20')), ((None, None,
'D9'),)
```

Súbor bude obsahovať riadky:

```
T20 T20
T20 20
T20 T20
5 1
T20 20
T20 1
T20 T20
1 T20
T20 T20
T19 T1
D14 5                ← D14 neplatný hod hráča 1
  T20                ← hráč 1 už nehodí v danom kole
X                    ← hráč 1 netrafil terč
X                    ← hráč 1 uzavrel leg, hráč 2 nehodil: žiadna medzera po X
D9
```

Poznámka: Príklady výstupných súborov nájdete v projekte v priečinku `sample_files`.

Úloha 4: `split_into_sets` (2 body)

Implementujte funkciu `split_into_sets`, ktorá ako vstup `throw_pairs` dostane zoznam hodov za jednotlivé kolá (ako výstup z funkcie `parse_throws()`), a vráti

trojrozmerný zoznam (zoznam zoznamov zoznamov), kde pôvodné dvojice reprezentujúce kolá sú rozdelené do setov a legov.

Výsledný zoznam teda bude mať tri úrovne:
1. vonkajší zoznam – zoznam reprezentujúci celý zápas;
2. stredné zoznamy – každý z nich reprezentuje jeden set;
3. vnútorné zoznamy – reprezentujú jeden leg.

Poznámka: Funkcia musí správne identifikovať koniec jednotlivých legov. Pri identifikácii konci setu vychádzajte z predpokladu, že každý set trvá dovtedy, kým jeden z hráčov nevyhrá tri legy – nezabudnite však na to, že v rámci setu začínajú hráči legy striedavo. Pre vyriešenie tejto úlohy stačí, ak správne identifikujete hráčov v rámci setu.

Úloha 5: `get_match_result` (1 bod)

Funkcia `get_match_result` dostane ako vstup štruktúrovaný zoznam kôl za jednotlivé sety resp. legy (ako výstup z funkcie `split_into_sets()`) a vráti jednu n-ticu (`tuple`), pričom tá obsahuje dve hodnoty:
1. prvá návratová hodnota je n-tica s dvomi číslami, ktoré reprezentujú počet vyhratých setov jednotlivých hráčov
2. druhá návratová hodnota je zoznam n-tíc, pričom každá n-tica obsahuje počet vyhratých legov jednotlivými hráčmi v jednotlivých setoch.

Príklad návratovej hodnoty (hráč 1 vyhral 3–1 na sety, hráč 2 vyhral iba tretí set):
```
((3, 1), [(3, 2), (3, 2), (1, 3), (3, 0)])
```

Poznámka: Funkcia musí správne identifikovať hráčov. Nezabudnite na to, že hráči otvárajú sety striedavo, a takisto sa striedajú pri otváraní legov v rámci jedného setu. Prvý set otvára vždy hráč 1, teda vieme povedať, že hráč 1 bude otvárať nepárne sety (prvý, tretí, piaty, siedmy, atď.) a hráč 2 bude otvárať párne sety (druhý, štvrtý, šiesty, atď.). Sety už budete mať rozdelené vo vstupnom zozname, avšak stále potrebujete správne namapovať identitu hráčov – to, že hráč hodí ako druhý v rámci legu neznamená, že je to hráč 2.

Úloha 6: `get_180s` (0,5 bodov)

Ak ste už niekedy videli zápas šípok v televízii, tak určite viete, s akým elánom hlásateľ oznamuje skóre 180, teda maximálne možné skóre dosiahnuteľné v rámci jedného kola. Funkcia `get_180s`, na základe vstupného štruktúrovaného zoznamu s formátom ako výstup z funkcie `split_into_sets()`, vráti počet hodených 180 jednotlivých hráčov za celý zápas. Návratová hodnota je teda dvojica (`tuple`) s dvomi celými číslami: počet hodených 180 hráča 1 a hráča 2.

Poznámka: Podobne ako v predošlej úlohe, aj tu je kľúčová správna identifikácia jednotlivých hráčov vo všetkých setoch a legoch.

Úloha 7: `get_average_throws` (1 bod)

Funkcia `get_average_throws` dostane ako vstupný parameter štruktúrovaný zoznam kôl za zápas (rovnaký formát ako výstup z funkcie `split_into_sets()`), a vráti priemerný počet hodených bodov jednotlivých hráčov za kolo pre každý jeden set ako zoznam dvojíc (`tuple`). Ak teda zápas pozostával z troch setov, výstup môže byť:

```
[(99.6, 118.79), (100.54, 108.41), (107.36, 101.42)]
```

Poznámka: Aj v tomto prípade musíte správne identifikovať hráčov, teda prvá hodnota z dvojice bude vždy zodpovedať priemernému počtu hodených bodov za kolo hráča 1 z pohľadu celého zápasu, bez ohľadu na konkrétny set/leg. Samozrejme neplatné hody sa nezarátavajú do priemeru. Pri hodnotení zadania bude akceptovaná aj malá odchýlka od očakávaných hodnôt, ktorá môže byť spôsobená prípadným zaokrúhľovaním.

Úloha 8: `generate_finishes` (2 body)

Implementujte funkciu `generate_finishes`, ktorá vygeneruje zoznam všetkých možných uzavretí s maximálne tromi šípkami pre istý počet bodov, ktorý dostane ako parameter `points`. Funkcia vracia zoznam všetkých možných uzavretí, pričom každé z nich je reprezentované ako zoznam (`list`) a hody sú zakódované rovnako, ako v ostatných úlohách. Pri uzavretí musí platiť, že posledný hod musí byť double, doublom sa ráta aj inner bullseye (teda posledný prvok bude tvaru `'D_'` alebo `'50'`). Ak sa pri danom počte bodov nedá hru uzavrieť jedným kolom, funkcia vracia prázdny zoznam. Najvyššia hodnota, pre ktoré existuje uzavretie je 170.

Poznámka: Za funkciu dostanete jeden bod, ak vygenerujete možné uzavretia, ďalší bod ak v zozname vrátite iba unikáty. Unikáty sa riešia v prípade trojice hodov, pričom platia nasledovné pravidlá:
- ak kombinácia obsahuje dva rôzne double hody, ich výmena sa berie ako dve rôzne kombinácie: teda pri počte bodov 12 trojica `['D3', '2', 'D2']` **a** `['D2', '2', 'D3']` **sa budú považovať za rôzne, keďže raz hráč končí** `'D2'`**, raz** `'D3'`**.**
- ak sa v kombinácii mení poloha single alebo triple hodnoty, neberie sa to ako iná kombinácia: teda kombinácie `['D3', '2', 'D2']` **a** `['2', 'D3', 'D2']` **sú považované za jednu jedinečnú kombináciu.**
- ak sa v kombinácii nachádzajú dva rovnaké hody, ich výmenou nevznikne nová kombinácia: teda kombinácia `['1', '1', 'D1']` **pre 4 body by sa mala vo výslednom zozname nachádzať iba raz.**

Napríklad možné kombinácie pre 4 body:
```
[['2', 'D1'], ['D1', 'D1'], ['1', '1', 'D1'], ['D2']]
```

Vaše riešenia môžete otestovať aj pomocou sady testov v súbore `assignment1_test.py`. Pri hodnotení vášho riešenia sa použijú podobné testy, avšak ich bude viac.

Približná dĺžka riešenia: *cca. 250 riadkov formátovaného kódu bez komentárov (jeden bod za 25 riadkov).*