Losovanie turnajov švajčiarskym systémom.

Výsledok práce zimného semestra:

- Zoznámenie sa s TRF formátom
- Odskúšanie dostupnej JavaFo aplikácie na štandardných vstupoch
- Analýza krajných situácii – ako zareaguje dostupná JavaFo aplikácia pri krajných situáciách, či dokáže aplikovať pravidlá (farby, ...)
- Naprogramovanie TRF interpretera, aby dal ako výstup dáta v čo najjednoduchšej dátovej štruktúre pre následné spracovanie mojím programom (mojím algoritmom - aplikáciou)
- JavaFo **nie je OPEN SOURCE** (kód nie je dostupný), mojím cieľom je spraviť tento kód dostupným (spraviť vlastnú aplikáciu), ktorá pri chybnom vstupe lepšie popíše chybu.
  Súčasný JavaFo pri chybnom vstupe vypíše chybu, nevypíše však typ chyby, ani dôvod je vzniku, **čo by som chcel zmeniť**.
- JavaFo je ***DETERMINISTICKÝ*** (vždy vygeneruje ten istý výstup, pri rovnakých vstupných údajoch)

Stránka na JavaFo: http://www.rrweb.org/javafo/aum/JaVaFo2_AUM.htm

TRF formát: https://www.fide.com/FIDE/handbook/C04Annex2_TRF16.pdf

Odkaz na TRF interpreter *(a všetky súbory, ktoré sa spomínajú v tomto texte)*:

https://github.com/XL202/JavaFo

Môj TRF interpreter aktuálne interpretuje a skontroluje správnosť (priebežne ho podľa potreby budem dopĺňať a vylepšovať):

- ✓ Či si dodržané medzery (voľné „columns“)
- ✓ Kontroluje (zatiaľ) len riadky 001 (t. j. riadky s údajmi o jednotlivých kolách)
- ✓ Číta údaje: player ID (rank) **col 5-8**, player name **col 15-47**, rating **col 49-52**, points **col 81-84**, jednotlivé odohraté hry (na jednu hru **col x2 – x5**, t. j. pre jednu hru je rezervovaná jedna desiatka)
- ✓ Kontroluje, či je spravené správne spárovanie: t. j. ak hral napr. 1 s 4, tak musel hrať aj 4 s 1 a zároveň museli mať 4 a 1 navzájom opačné farby a navzájom opačné výsledky, resp. remízu.
- ✓ Kontroluje dodržiavanie pravidiel, čo sa týka farieb: nesmú byť 3 rovnaké farby po sebe a celkový rozdiel farieb nesmie byť väčší ako 2 a menší ako -2. (pod +x sa rozumie x-krát **w (white)** a pod -x sa rozumie x-krát **b (black)**).
  *(rozdiel sa počíta ako **w**-**b**)*
- ✓ Kontroluje pravidlo, že pre každú dvojicu v turnaji platí pravidlo, že sa stretli len raz (t. t. j. neexistuje dvojica taká, že hrali spolu v turnaji viac ako 1-krát)
- ✓ Posledné kolo (ak je pomocou **XXR #** v samostatnom riadku uvedený maximálny počet kôl) sa uplatňuje výnimka na pravidlá z farieb.
- ✓ Vracia potrebné údaje pre generátor ďalšej dvojice (pre každého hráča – ako dátovú štruktúru „map“ (HashMap) mapu):
  - s kým už daný hráč hral
  - s kým ešte môže hrať (s kým ešte nehral)
  - Koľko bolo posledných rovnakých farieb
  - aký je rozdiel farieb
  - aktuálny počet bodov
  - Hráčov RATING

Pri zoznamovaní sa s .TRF formátom som využíval RandomTournamentGenerátor a pri overovaní krajných vstupov som využíval generátor v JavaFo:

Príkazy pre spúšťanie javafo.jar:

Radnom generátor:

```
java -ea -jar PATH/javafo.jar -g [podmienky] -b -o [do súboru]
```

```
java -ea -jar javafo.jar -g PATH/S01.txt -b -o PATH/trn.trf
```

!!! zistená nevýhoda radnom generátora: v prípade zadania pravdepodobnosti na remízu na 0% resp. 100% sa vyskytla nejaká remíza, resp. niektorý výsledok nebola remíza !!!

V súbore ***random100.trf*** je vygenerovaný turnaj s nastaveným parametrom:

DrawPercentage=100 v ***S01.txt***

všetky hry však nekončia remízou.

V súbore ***random0.trf*** je vygenerovaný turnaj s nastaveným parametrom:

DrawPercentage=0 v ***S01.txt***

existujú aj hry, ktoré skončili remízou.

PATH/ => cesta do priečinka, kde je javafo.jar alebo treba príkaz spúšťať z priečinka, kde je javafo.jar

PATH/ => cesta do priečinka, kde je daný súbor / kde má byť vytvorený súbor alebo bude súbor hľadaný/vytvorený v priečinku, kde je javafo.jar

S01.txt – podmienky pre RandomTournamentGenerátor

Generátor dvojíc:

```
java -ea -jar PATH/javafo.jar -r [file_from] -p [file_to]
```

```
java -ea -jar PATH/javafo.jar -r PATH/vstup2_6.trf -p PATH/out_2.txt
```

V prípade, že riadok **XXR** nie je uvedený, do output súboru sa vygeneruje posledné odohrané kolo (pokiaľ nejaké existuje – inak sa vyhodí chyba).

V prípade, že je uvedený počet kôl, ktoré sú už aktuálne odohrané (napr. pri piatich kolách **XXR 5**, tak sa vygeneruje tiež posledné kolo)

V prípadne ak je **XXR #** menej ako aktuálny počet odohraných kôl, tak vyhodí chybu (že počet kôl presiahol maximum).

Experimentovaním som zistil že, keď sú dve rovnaké kola za sebou, tak nevyhlási chybu. (viď súbor ***vstup1_4_chyba.trf***. Očakával by som, že JavaFo vyhodí chybu pre rovnaké páry.

Odskúšanie JavaFo na krajných prípadoch (zadanie aj v súbore: ***zadanie.txt***):

VSTUP1:

4 (6) hráči 3 kolá - 3. kolo sa nedá urobiť, 2 kolá sa už odohrali.

Cieľom je overiť obe pravidlá na farby:

1) nemôže sa vyskytnúť 3x po sebe rovnaká farba.
2) Nemôže my rozdiel farieb väčší ako 2 (t. j. rozdiel musí byť z intervalu <-2; 2>)

Prvý pokus je v súbore ***vstup1_4.trf***.

v poslednom treťom kole nie je možné spraviť legálne párovanie, pretože po 2. kole je už jasné, aké musí byť 3.kolo (aké musia byť páry aj farby). Vynútené páry v 3.kole sú 4-1 a 3-2, ale vynútené farby pre tieto páry už nie sú legálne.

V páre 4-1 musia byť dve rôzne farby (nemôžu obaja hráči hrať s rovnakou farbou), čo je spor s vynútenými farbami (obaja "w").

V páre 3-2 musia byť dve rôzne farby (nemôžu obaja hráči hrať s rovnakou farbou), čo je spor s vynútenými farbami (obaja "b").

JavaFo zareagovala vyhodením chyby - nie je možné spraviť párovanie s dodržaním pravidiel párovania (zareagovala správne)

```
java -ea -jar javafo.jar -r vstup1_4.trf -p out1_4.txt
```

(nevytvorí sa žiadny output súbor)

Tento vstup (***vstup1_4.trf***) overil prvé pravidlo na farby (nemôže sa vyskytnúť 3x po sebe rovnaká farba).

Na overenie 2. pravidla musí byť v turnaji minimálne 5 kôl, budem sa snažiť vynútiť situáciu WWBWW resp. BBWBB (rozdiel bude +3, resp. -3)

V súbore ***vstup1_6_1.trf*** bude v poslednom (5.kole) vynútená dvojica 1-2 – tá však podľa pravidiel nie je možná bez porušenia pravidiel 1 alebo 2.

V súbore ***vstup1_6.trf*** bude v poslednom (5.kole) vynútená dvojica 1-2 – tá však podľa pravidiel nie je možná bez porušenia pravidla 2.

JavaFo zareagovala vyhodením chyby - nie je možné spraviť párovanie s dodržaním pravidiel párovania (zareagovala správne)

```
java -ea -jar javafo.jar -r vstup1_6.trf -p out1_6.txt
```

(nevytvorí sa žiadny output súbor)

```
java -ea -jar javafo.jar -r vstup1_6_1.trf -p out1_6_1.txt
```

(nevytvorí sa žiadny output súbor)

Tieto vstupy (***vstup1_6.trf*** a ***vstup1_6_1.trf***) overili druhé pravidlo na farby (rozdiel by bol +3 resp. -3).

VSTUP2

2 kolá pred koncom má hráč na výber 2 súperov (1 prirodzený na základe bodov, 2. ako krajná možnosť) - dá sa urobiť

Kvôli farbám v poslednom kole, sa musí vybrať v predošlom kole tá krajná možnosť, lebo len s ňou bude sedieť farba v poslednom kole.

```
java -ea -jar javafo.jar -r vstup2_8.trf -p out2_8.txt
```

JavaFo vygeneruje výstup

4

3 1

2 4

8 5

6 7

Komentár v súbore ***vstup2_8_comment.txt***

Týmto vstupom som chcel overiť, či JavaFo myslí aj kroky dopredu (čí generuje aktuálne kolo s prihliadnutím aj na ďalšie kolá (aby sa dali spraviť), alebo generuje iba aktuálne kolo a nerieši ďalšie kolá.

*aby sa neuplatnila výnimka v poslednom kole, zvýšim XXR 7 na XXR 8

JavaFo nerieši čo bude v ďalšom kole, preto si vybral možnosť tak, akoby ďalšie kolo už nebolo. Navyše pomocou XXR sa určuje len maximálny počet kôl, nie povinný počet.

Keďže JavaFo vygenerovala prirodzený výstup, a kvôli vygenerovanému prirodzenému výstupu pre aktuálne kolo sa nebude dať legálne spárovať posledné kolo (JavaFo správne vyhodila chybu, že sa to nedá), existujú 2 možnosti:

- Aktuálne správanie sa zachová (t . j. vyhodí sa chyba o tom, že sa nedá legálne spárovať nasledujúce kolo)
- Vygeneruje sa nasledujúce kolo tak, že, páry, ktoré by porušovali pravidlá párovania, kvôli ktorým by JavaFo vyhodila chybu, nebudú v danom kole hrať (systém im nepridelí spoluhráča, t. j. budú mať „nepridelenie systémom: **`0000 – U`**“.

V prípade aplikácie 2. možnosti by sa však aj v prípade legálneho spárovania dalo spraviť „nepridelenie systémom“ vtedy, ak by legálne spárovanie bolo príliš neprirodzené (hrali by najlepší hráči s najhoršími (rozdiel bodov by bol veľký (dvojice hráčov by boli neprirodzené)).

Nepridelenie systémom by generátor mal dávať primárne tým hráčom, ktorý už veľmi zápas neovplyvnia: t. j. hráčom, ktorý majú najmenší počet bodov.

Môj generátor by teda mal pracovať tak, že sa bude snažiť spárovať najprv hráčov, ktorí majú najviac bodov (tí ktorí môžu najviac ovplyvniť zápas), až potom tých čo majú najmenej bodov, prípadne tým, čo majú najmenej bodov systém pridelí „nepridelenie systémom: **0000 – U**“.

Očakávam, že nepridelenie systémom bude program hlavne riešiť pri malých vstupoch, lebo tam je menej možných kombinácii a je teda aj pravdepodobnejšie, že skôr by sa vyskytol problém s farbami.

JavaFo už prideľuje „nepridelenie systémom: **0000 – U**“ v prípade nepárneho počtu hráčov. (t. j. keďže páry tvorí párny počet hráčov a jeden bude vždy nespárovaný – samozrejme sa snaží dať **0000 – U** hráčovi s najmenším počtom bodov). Počet **0000 – U** pri nepárnom počte hráčov teda bude nepárny a počet pri párnom počte hráčov , bude párny v každom kole.

VSTUP3

Hráč ma na výber dvoch súperov, keď si vyberie prirodzeného, tak zvyšných sa mu nepodarí spárovať, preto si musí vybrať krajnú možnosť.

JavaFo sa zachoval správne, viď komentár v súbore ***vstup3_6_coment.trf***. JavaFo si vybral krajnú správnu možnosť pred prirodzenou nesprávnou možnosťou.