"Słownik" używany w ASR wygląda następująco:

```
ort_word [Disp Word] f o n e m i z a c j a
```

na przykład:

```
chleb [chleb] h l e p
pdf_y [PDF-y] p e d e e f y
giro_d_italia [Giro d'Italia] drz i r o d i t a l j a
```

Cechy poszczególnych elementów:

- **ort word:**
  - każde posiada w modelu języka przypisane prawdopodobieństwo wystąpienia w tekście, oparte (w uproszczeniu) na częstości występowania w korpusie
  - musi być unikalne w obrębie słownika
  - nie może zawierać spacji, może zawierać `_`, który zgodnie z przyjętą konwencją zastępuje łącznik, apostrof lub ew. spację, może zawierać cyfry
  - "`pdfy`" i "`pdf_y`" są traktowane jako osobne ort wordy
  - standardowo wszystkie są w lowercase, ale są obsługiwane case-sensitive, więc "`facebook`" i "`Facebook`" są traktowane jako osobne ort wordy

- **disp word:**
  - domyślnie identyczne z ort word, ale można je dowolnie ręcznie edytować
  - nie musi być unikalne
  - obsługiwany jest tylko jeden disp word dla jednego ort word, inne są pomijane (case-sensitive, "`facebook`" i "`Facebook`" to osobne disp wordy)
  - może zawierać spacje, wielkie litery, inne znaki

- **fonemizacja:**
  - generowana automatycznie dla każdego ort word, z użyciem modułu g2p
  - nie musi być unikalna (ale nadmierna ilość homofonów znacznie obciąża ASR)
  - w przypadku modeli fonemowych - jedna para ort word disp word może mieć wiele fonemizacji, w przypadku modeli literowych – jedna para, jedna fonemizacja

Słownik może zawierać nawet do 1 miliona unikalnych ort_wordów.
Stosowane są UNIXowe końce linii i kodowanie UTF-8.

Problem

W wyniku łączenia słowników przygotowywanych przez różne osoby w różnym czasie, lub generowanych automatycznie z weryfikowanymi ręcznie, pojawiają się różnego rodzaju **problemy – niezgodności z opisanymi powyżej regułami tworzenia słowników**.

W załączonym pliku problem.dict znajduje się taki właśnie „zabałaganiony" słownik (fikcyjny, rzecz jasna).

Przeanalizuj słownik i napisz skrypt w Pythonie, który:

- jako argument przyjmie plik słownika i ew. inne informacje
- korzysta z argparse
- **wygeneruje plik/pliki tekstowe zawierające problematyczne linie, z podziałem na typy błędów**

Zaproponuj metodę i kolejność działania podczas poprawiania takiego słownika wykorzystując pliki wygenerowane przez skrypt. Na pewno będzie to (przynajmniej częściowo) „ręczna" weryfikacja, ale opisz ten proces choć trochę bardziej szczegółowo.

Jakie widzisz problemy, które mogą pojawić się w słowniku?
Czy i w jaki sposób można by zautomatyzować poprawianie takiego słownika? Na których etapach?