Program po podaniu mu tekstu na wyjście zwraca 3-elementową krotkę której pierwszym elementem jest kod, drugim None albo co ma się zmienić, trzecim None albo na co ma się zmienić np. (A-1, None, None), (D-5, kolor, czerwony)

Gramatyka:

Słuchanie zaczyna się gdy usłyszymy *słowo aktywacji* (akw, domyślnie „*dom*”)

Polecenie musi zakończyć się *słowem kończącym* (ekw, domyślnie „*proszę*”)

Polecenie musi zaczynać się w jeden z 3 sposobów:

Jedno wyrazową **komendą** (słowa które zaliczają się jako **komendy** są w pliku parameters.py on/off/up…)

Po rozpoznaniu komendy znajduje się nazwa obiektu np.:

*dom* **załącz** światło w kuchni *proszę*

*dom* **wyłącz** światło w kuchni *proszę*

Komendą typu ***set***, komenda typu ***set*** pozwala na ustawienie pewnej rzeczy na konkretną wartość np.:

*dom* ***ustaw*** kolorświatło w kuchni nazielony *proszę*

*dom* ***zmień*** głośność telewizora w dużym pokoju na 16 *proszę*

Konkretną wartość musi być poprzedzona słowem na lub innym które dodamy do zmiennej for\_kw

Lub od razu nazwą przedmiotu np.:

Dom światło w kuchni *proszę*

Taka komenda jest interpretowana jako komenda toggle

Odległość edycyjna

max\_ed:

max edit distance

Float który mowi o tym jak duża może być odległość edycyjna w porównaniu do długości słowa. Przy max\_ed = 0.3 maksymalna różnica edycyjna między naszym słowem (s) a **komendą (**lub *słowem aktywacji*) jest równa:

ceil(max\_ed \* len(s))

np.:

usłyszane słowo: s = „tom” (len(s) == 3)

tolerancja: ceil(0.3 \* 3) == 1

Uznajemy że usłyszeliśmy słowo aktywacji *„dom”*

usłyszane słowo: s = „przełącz” (len(s) == 8)

tolerancja: ceil(0.3 \* 8) == 3

Nie uznajemy słowa s jako **„wyłącz”** ponieważ odległość edycyjna tych dwóch wyrazów jest równa 4

Mamy do wyboru dwie implementacje algorytmu znajdującego odległość edycyjną. Jedną implementację wykonałem ja (jest ona umieszczona w pliku edit\_dist.py umożliwia zmianę funkcji dystansu), drugą trzeba zainstalować. Możemy je zmienić w pliku structures.py.

House

Klasa House służy do przechowywania obiektów którymi chcemy operować, sam przypisuje im odpowiednie id zaczynając od „A” do „Z” po czym „ZA” itd.

Do instancji klasy House dodajemy obiekt przekazując mu nazwę obiektu, jeżeli chcemy aby jeden obiekt miał dwie nazwy możemy dodatkowo przekazać obj\_id który jest taki sam jak wcześniej dodanego obietku np.:

house = House()

house.add\_obj(„światło w dużym pokoju”) ID = „A”

house.add\_obj(„światło w kuchni”) ID = „B”

house.add\_obj(„światło”, „A”) ID = „A”

house.add\_obj(„budzik”) ID = „C”