

```
1 import re
2 from importlib.resources.readers import remove_duplicates
3
4
5 class Data:
6
7     # list of possible type of questions that will be used in conditional statements [non-exhaustive]
8     possibleList = [{"capital", "distanc", "weather", "movi", "forecast", "cit", "length", "climat", "humidit", "director", "actor"}, # direct-answer question
9                     ["exercis", "diet", "cook", "workout", "routin", "gym", "activit", "food", "nutri", "wellness", "recipi", "fitnes"], # health-related questions
10                    ["calendar", "remind", "task", "schedul", "event", "deadlin", "project", "checklist", "alert", "notif", "organ"], # productivity questions
11                    ["scor", "gam", "jok", "song", "challeng", "puzzl", "music", "lyric", "match", "adventur", "humor"], # entertainment questions
12                    ["plu", "minus", "multipl", "divid", "formula", "concept", "ratio"], # mathematical questions
13                    ["pric", "mean", "fact", "happen", "latest", "explain"], # knowledge-building question
14                    ["best", "advic", "help", "tip", "plan"]] # advice-seeking questions
15
16     # list of possible cities that the user might reference [non-exhaustive]
17     specificPlaceList = ["Los Angeles", "Chicago", "San Francisco", "Miami", "Austin", "Las Vegas", "Paris", "London", "Tokyo", "Sydney", "Rome", "Barcelona",
18                         "Berlin", "Dubai", "Toronto", "Seoul", "Bangkok", "Mexico City", "Riverside", "Cape Town", "California", "Florida", "Texas", "New York", "Nevada", "Hawaii", "Colorado", "
19                         Alaska", "Arizona",
20                         "Utah", "Illinois", "Michigan", "Washington", "Georgia", "North Carolina", "Tennessee", "South Carolina", "Oregon", "New Jersey", "Virginia", "United States", "Canada", "
21                         United Kingdom",
22                         "France", "Italy", "Spain", "Mexico", "Germany", "Australia", "Brazil", "Japan", "India", "South Korea", "Thailand", "South Africa", "China", "Russia", "Egypt", "Argentina", "
23                         New Zealand", "Pakistan"]
24
25     # list of possible pop-culture references [non-exhaustive]
26     specificPopCultureList = ["The Avengers", "Star Wars", "The Matrix", "Harry Potter", "Jurassic Park", "Titanic", "The Godfather", "Pulp Fiction", "Back to the Future", "The
27     Lion King", # movies
28     "Apple", "Nike", "Tesla", "Coca-Cola", "McDonald's", "Amazon", "Google", "Adidas", "Disney", "Microsoft", # brands
29     "Friends", "Game of Thrones", "The Office", "Stranger Things", "Breaking Bad", "The Simpsons", "The Mandalorian", "The Crown", "The Walking Dead", "Westworld", # tv
30     shows
31     "The Beatles", "Beyonce", "Kanye West", "Taylor Swift", "Elvis Presley", "Michael Jackson", "Ariana Grande", "Drake", "Lady Gaga", "Eminem", # artists
32     "Super Mario Bros.", "Minecraft", "Fortnite", "The Legend of Zelda", "Call of Duty", "Grand Theft Auto", "Pokémon", "League of Legends", "FIFA", "The Witcher", # video
33     games
34     "Rolls-Royce", "Ferrari", "Lamborghini", "Porsche", "Maserati", "Bentley", "Aston Martin", "Bugatti", "McLaren", "Mercedes-Benz"] # automotive brands
35
36     # method that removes accidental duplicates found by regular expression
37     def __remove_duplicate(self, list):
38         accList = []
39         for i in list:
40             if i not in accList:
41                 accList.append(i)
42         return accList
43
44     # uses Python Regular Expressions to derive key data in a structural format, replicating a basic version of Natural Language Processing
45     # "QT" = Question Type && "I" = Identifier
46     def parsedData(self, userInput):
47         question_type = self.__remove_duplicate(re.findall(r"(what|who|why|where|when|how|will|can|play|should|is)", userInput))
48         identifiers = self.__remove_duplicate(re.findall(r"(capital|best|cit|length|climat|humidit|director|actor|task|schedul|event|deadlin|project|checklist|alert|notif|organ|advic|help|
49         tip|distanc|plan|weather|forecast|latest"
50         r"|happen|movi|exercis|song|diet|workout|explain|routin|gym|activit|food|nutri|wellness|recipi|fitnes|calendar|remind|cook|scor|pric|mean|plu
51         |ratio|minus|multipl|divid|"
52         r"|jok|gam|fact|formula|concept|challeng|puzzl|music|lyric|match|adventur|humor)", userInput))
53
54     # returns error if either one of the variables above is empty, else, normal dictionary returned
55     if len(question_type) < 1 or len(identifiers) < 1:
56         issue = {"QT": "ERROR", "I": "ERROR"}
57         return issue
58     else:
59         result = {"QT": question_type, "I": identifiers}
60         return result
61
62     # takes the original sentence inputted by the user and then removes suffixes; local change not global
63     def stemWord(self, userInput):
64         return re.sub(r'\b(?:\b|s|se|ication|ization|isation|ized|ised|ied|ous|y|ies|tion|ent|ents|er|ers)\b', "", userInput)
```