```python
1  from data_class import Data
2  from loadingModule_2 import loadingAnimation
3  import sys
4  import string
5  import time
6
7  # method that returns the question-type like 'what', 'when', where' etc...
8  def printAllQuestionType(processed_data):
9      if len(processed_data.get("QT")) == 1:
10         return "'" + processed_data.get("QT").__getitem__(0) + "'"
11     else:
12         result = "mix of "
13         try:
14             for c in range(len(processed_data.get("QT")) - 1):
15                 result += "'" + processed_data.get("QT").__getitem__(c) + "', "
16             result += "and '" + processed_data.get("QT").__getitem__(len(processed_data.get("QT")) - 1) + "'"
17         except IndexError:
18             print("\nI had an issue processing your query. Please re-run the program and rephrase your sentence.")
19             sys.exit()
20     return result
21
22 # method that returns any possible places that the user mentioned in their original statement ~ improves specific
   contextualization
23 def printPossibleMentions(saved_input, list):
24     matches = [place for place in list if place.lower() in saved_input.lower()]
25
26     if len(matches) == 1:
27         return matches[0]
28     elif matches:
29         return ", ".join(matches)
30     else:
31         return ""
32
33 # method that checks if the given word is in the possibleList matrix to ensure that it can proceed with "what is"
34 def validateItemAbsent(word, data_instance):
35     for q in range(len(data_instance.possibleList)):
36         for w in range(len(data_instance.possibleList[q])):
37             for e in word:
38                 if e == data_instance.possibleList[q][w]:
39                     return False
40     return True
41
42 # introductory instructions and terms/conditions
43 print("QUESTION ANALYZER: \n1.) Enter a question that you might ask a Google Assistance, Alexa, Siri, Cortana, etc..."
44     "\n2.) The system will process your response.\n3.) It will give you it's understanding of the question by categorizing it
   using basic Natural Language Processing (NLP) algorithm.\n")
45 print("This is a first step taken to complete a part of 'robotics data interpretation' and will be built upon modularly.\n")
46 print("This model is still in BETA; some questions might not be recognizable by the system. More updates will be rolling out soon
   . [Version: 2.0]\n\n")
47 user_input = input("Ask me anything: ")
48 data_instance = Data()
49 rowCategory = None
50
51 # timer started - to be used to calculate how long this program was used
52 start_time = time.time()
53
54 # loop helps the user continue asking more questions for system categorization
55 while user_input != "stop":
56     saved_input = user_input
57     user_input = user_input.lower()
58     processed_data = data_instance.parsedData(data_instance.stemWord(user_input), saved_input)
59
60     # ensures there are no error, else, redirect the issue
61     if processed_data.get("I") != None:
62         loadingAnimation()
63         for row in range(len(data_instance.possibleList)):
64             for column in range(len(data_instance.possibleList[row])):
65                 for a in processed_data.get("I"):
66                     if data_instance.possibleList[row][column] == a:
67                         rowCategory = str(row)
68         if rowCategory == None and not saved_input.__contains__("what if") and not saved_input.__contains__("what is"):
69             user_input = input("\nI wasn't able to understand your question. I can comprehend the question type but I couldn
   't find any identifiers that can help process this query.\nTry again or type 'stop': ")
70         else:
71             print(f"\n\n\nI understand that you are trying to ask a question that starts with a {printAllQuestionType(
   processed_data)} and I am supposed to give a(n) ")
72             place_mentions = printPossibleMentions(saved_input, data_instance.specificPlaceList)
73             pop_culture_mentions = printPossibleMentions(saved_input, data_instance.specificPopCultureList)
74
75             # list of possible type of questions extracted from user input [non-exhaustive]
76             if saved_input.__contains__("what if") or saved_input.split()[0] == "if":
77                 if pop_culture_mentions != "":
78                     print("thoughtful, speculative answer based on logical reasoning, established facts, and potential
   scenarios of " + str(pop_culture_mentions) + ".")
79                 elif place_mentions != "" and pop_culture_mentions == "":
80                     print("thoughtful, speculative answer based on logical reasoning, established facts, and potential
   scenarios of " + str(place_mentions) + ".")
81                 else:
82                     print("thoughtful, speculative answer based on logical reasoning, established facts, and potential
   scenarios.")
```

```
83              elif (saved_input.__contains__("what is") and (validateItemAbsent(data_instance.correctedWord, data_instance)
        or len(saved_input.split()) == 3)) or saved_input.__contains__("definition"):
84                  clean_text = saved_input.translate(str.maketrans('', '', string.punctuation))
85                  if clean_text.split().__getitem__(len(clean_text.split()) - 1).isupper():
86                      clean_text = clean_text.upper()
87                  print("definition and explanation for " + clean_text.split().__getitem__(len(clean_text.split()) - 1) +
        ".")
88              elif rowCategory == "0":
89                  if place_mentions != "" and pop_culture_mentions != "":
90                      print("clear and succinct response providing general insights into situations, activities, individuals
        , or weather conditions and forecasts for locations like " + str(place_mentions) + " and Iconic References like " + str(
        pop_culture_mentions) + ".")
91                  elif place_mentions != "":
92                      print("clear and succinct response providing general insights into situations, activities,
        individuals, or weather conditions and forecasts for locations like " + str(place_mentions) + ".")
93                  elif pop_culture_mentions != "":
94                      print("clear and succinct response providing general insights into situations, activities,
        individuals, or weather conditions and forecasts for Iconic References like " + str(pop_culture_mentions) + ".")
95                  else:
96                      print("clear and succinct response providing general insights into locations, situations, activities,
        individuals, or weather conditions and forecasts, grounded in established facts.")
97              elif rowCategory == "1":
98                  print("insightful, evidence-based response addressing personal or health-related inquiries about diet
        , lifestyle, and well-being.")
99                  if place_mentions != "" or pop_culture_mentions != "":
100                     print("You mentioned places and/or Iconic References like, yet I don't see any relevancy of these
        mentions to the personal or health-related inquires you asked.")
101             elif rowCategory == "2":
102                 if place_mentions != "" and pop_culture_mentions != "":
103                     print("productivity-focused response designed to enhance health, work, or performance efficiency,
        based on logical reasoning, proven strategies, and relevancy to the following places: " + str(place_mentions) + " and Iconic
        References: " +
104                           str(pop_culture_mentions))
105                 elif place_mentions != "":
106                     print("productivity-focused response designed to enhance health, work, or performance efficiency,
        based on logical reasoning, proven strategies, and relevancy to the following places: " + str(place_mentions) + ".")
107                 elif pop_culture_mentions != "":
108                     print("productivity-focused response designed to enhance health, work, or performance efficiency,
        based on logical reasoning, proven strategies, and relevancy to the following Iconic References: " + str(pop_culture_mentions
        ) + ".")
109                 else:
110                     print("productivity-focused response designed to enhance health, work, or performance efficiency,
        based on logical reasoning and proven strategies.")
111             elif rowCategory == "3":
112                 if place_mentions != "" and pop_culture_mentions != "":
113                     print("engaging, mood-enhancing response tailored to improving your leisure and overall well-being,
        based on entertainment preferences and refined for the following places: " + str(place_mentions) + " and Iconic References: " +
114                           str(pop_culture_mentions))
115                 elif place_mentions != "":
116                     print("engaging, mood-enhancing response tailored to improving your leisure and overall well-being,
        based on entertainment preferences and refined for the following places: " + str(place_mentions) + ".")
117                 elif pop_culture_mentions != "":
118                     print("engaging, mood-enhancing response tailored to improving your leisure and overall well-being,
        based on entertainment preferences and refined for the following Iconic References: " + str(pop_culture_mentions) + ".")
119                 else:
120                     print("engaging, mood-enhancing response tailored to improving your leisure and overall well-being,
        based on entertainment preferences.")
121             elif rowCategory == "4":
122                 print("concise or detailed mathematical response addressing a concept or calculation, based on
        established principles and logical reasoning.")
123                 if place_mentions != "" or pop_culture_mentions != "":
124                     print("You mentioned places and/or Iconic References, although I am not sure the relevancy of this in
        the context of mathematical calculation, therefore, I am disregarding them.")
125             elif rowCategory == "5":
126                 if place_mentions != "" and pop_culture_mentions != "":
127                     print("informative response aimed at enhancing your understanding of key details related to current or
        past events and information for places like " + str(place_mentions) + " and Iconic References like " + str(pop_culture_mentions
        ) + ".")
128                 elif place_mentions != "":
129                     print("informative response aimed at enhancing your understanding of key details related to current or
        past events and information for places like " + str(place_mentions) + ".")
130                 elif pop_culture_mentions != "":
131                     print("informative response aimed at enhancing your understanding of key details related to current
        or past events and information for Iconic References like " + str(pop_culture_mentions) + ".")
132                 else:
133                     print("informative response aimed at enhancing your understanding of key details related to current or
        past events and information, grounded in factual accuracy and context.")
134             elif rowCategory == "6":
135                 if place_mentions != "" and pop_culture_mentions != "":
136                     print("practical and insightful advice designed to offer you the most effective guidance for better
        decision-making and results regarding the following places: " + str(place_mentions) + ".")
137                     print("However, you mentioned Iconic References like " + str(pop_culture_mentions) + ", yet I
        disregarded them due its irrelevancy to advice giving.")
138                 elif place_mentions != "":
139                     print("practical and insightful advice designed to offer you the most effective guidance for better
        decision-making and results in the following places: " + str(place_mentions) + ".")
140                 elif pop_culture_mentions != "":
141                     print("practical and insightful advice designed to offer you the most effective guidance for better
        decision-making and results.")
142                     print("However, you mentioned Iconic References like " + str(pop_culture_mentions) + ", yet I
```

```
142  disregarded them due its irrelevancy to advice giving.")
143                          else:
144                              print("practical and insightful advice designed to offer you the most effective guidance for better
         decision-making and results.")
145                          user_input = input("\nIs my understanding right? Type 'Y' for Yes or 'N' for No: ")
146                          print("Glad I am doing it right. Data has been noted!") if user_input == "Y" else input("In what way should I
         have interpreted the response: ")
147                          print("Thank you for your feedback!\n")
148                          user_input = input("\nAsk me anything (or type 'stop' to end): ").lower()
149          else:
150              user_input = input("\nThis question is unrecognizable. Try again or type 'stop': ").lower()
151
152  # post-program time display and feedback system
153  end_time = time.time()
154  total_time = int(end_time - start_time)
155  min = str(int(total_time / 60))
156  sec = f"0{int(total_time % 60)}" if (total_time % 60) < 10 else str(int(total_time % 60))
157  user_input = input(f"Now that you have used the program for {min}:{sec}, let me know how well I did: ")
158  print("Your feedback will help in improving this QuerySense Model. Thank you.")
```