Assignment 1 - User Interaction and Collections

Siri (by Apple) and Alexa (by Amazon) , are two of the most famous chatbots. They have answers to (pretty much) all your questions and they even undertake more complicated stuff like playing songs and ordering products from the internet.

For this assignment you will have to implement your own command-line chat bot that responds to basic questions!
As a starting point, you are given 3 classes:

a) The Main class - the class responsible for running the program
b) A Question class - the class representing questions and their answers.
c) The ChatBot class - the class of your chatbot
d) questions.txt - the file with the questions and answers

Problem

You are given the following code in Main.java. This code creates a new object of the ChatBot class using the constructor and initialises the name of the bot. It asks the user to enter her/his name using the Scanner class and saves the user response in a String called username. Further, it prints the user response at the screen.

```
ChatBot mybot = new ChatBot("MariaBot");
System.out.println("hello! my name is " + mybot.name);
System.out.println("What is your name? ");
Scanner scanner = new Scanner(System.in);
String username = scanner.nextLine();
System.out.println("hello!"+ username +" ask me a question!" );
```

Your job is to create a dynamic chatbot that has predefined questions and responses for the user.

Part1 - Implement Question.java

Create a new class **Question** and save it into a file **Question.java** . The file shall represent a question. It must include a number to identify a question, an answer, and an array of keywords that this question responds to.
When the user asks “What is the capital of Cyprus” the chatbot shall reply “The capital is Nicosia”.

For example:

questionID	1
answer	“The capital is Nicosia”
keywords	{“Cyprus”,”capital”}

Part 1.1 Define the appropriate attributes for the class **Question**. Think what **type** each attribute shall be.

Part 1.2 Create the **constructor** of the class Question. The constructor shall take as arguments the questionID, the answer and the keywords.

Part2 - Implement ChatBot.java

Your Chatbot will be able to respond to predefined questions. In ChatBot.java you need to implement the appropriate functionality to store **questions** in an appropriate data structure.

The class already contains the name of the ChatBot and a Set to save the **Question** objects.
public String name;
Set<Question> questions;

Part 2.1 Implement a method to add a question object to the set of Questions. The method shall receive the questionID, the keywords array and the answer and add it to the questions object.

```
public void addQuestion( ..  id, .. keywords, … answer){
      questions.add(new Question ( … ) ;

  }
```

Part 2.2 Implement a method to read **read questions and answers from a file**

Questions shall be stored in a text file “questions.txt” . Each line of the file represents a question and contains the **questionID**, the symbol “**-**” which is used as a delimiter the **keywords** separated by comma, the symbol “**-**” and the **answer**

Id **-** keyword1,keyword2,keyword3… **-** answer

4-favorite,football,team,Cyprus-My favorite football team is Anorthosis
5-capital, Cyprus-The capital of Cyprus is Nicosia
6-capital,Italy-The capital of Italy is Paris

Your method shall read accept as argument **the path of the file**, and save into Question the set **of Question** .

```
public void  readQuestionsfromFile (String path) {
        …
        // read file - for each line split string, create Question and add it into the set

        //addQuestion(... )
}
```

HINT :

1. use **String words[] = str1.split(" ");** to split a string using " " (space as a delimiter). You can use any delimiter you like.
2. use **str1.equals(str2)** to compare two strings
3. use **str1.trim()** to remove whitespaces at the beginning and end of the string.

Part 2.3 What shall happen when you create a new ChatBot? Implement a **constructor** for your chatbot that **receives** the path of the question files, reads the questions from the file, and **saves** them to the questions attribute of the class ChatBot.

Part 2.4 Implement a method that answers the question of the user, using the questions set contained in the Chatbot object. The method shall return the answer of the question, if all the keywords are contained into the users' question.

For example, when the user asks
"What is your **favorite football** team?"
The bot shall respond
"My favorite football team is Manchester"

Because it has matched the keywords favorite and football.

```
public String answerQuestion(String question){
        …

}
```

Part 3 - Interact with the user!

If you came thus far, well done! Now, in Main.java you need to add functionality to allow the user to use your bot.

Part 3.1 Write the appropriate code in your **Main.java** to read the question of the user, call the appropriate method of the chatbot to answer the question and print the answer to the user.

Part 3.2 Add a response for your chat bot in case it doesn't know the answer. Such as "**Not sure, any other questions?**"

Part 4 - Multiple keywords!

Excellent! You are one of the few that came this far! But let's admit that the bot is not the smartest one.. Using only one question keyword makes things generic and multiple questions can be wrong! For example if we have

Question keyword: **capital**
Response: The capital of Cyprus is Nicosia

Then anytime the user asks for the capital of any other country than Cyprus the chatbot will respond wrongly. Implement the appropriate code to respond correctly to the following questions.

4 - favorite,football,team,Cyprus - My favorite football team is Anorthosis
5 - capital, Cyprus - The capital of Cyprus is Nicosia
6 - capital ,Italy - The capital of Italy is Paris

Submission

Copy paste your files into the appropriate text boxes, as described in Assignment 1 on Blackboard.

Marking

Implementation of ChatBot.java	**30%**
Implementation of Question.java	**10%**
The chatbot can answer one question	**20%**
The chatbot can answer multiple questions	**20%**
Your program behaves correctly after adding new questions to the file	**20%**