Wednesday, April 25, 2018

Ariel: chat bot project report

Minh Hien Pham - sc17mp

This report include a brief explanation of the (planned) design and testing of the chat bot, for module COMP1921, programming project 2, project choice number 1 - making a simple chat bot.

1. Summary

I have chosen project choice 1- to make a simple chat bot.

This chat bot will work as an encyclopedia, focusing on information about different countries including the continent where it's at, its capital city and population. The bot will also be able to give random interesting facts on the same topic, as well as some information about itself. It will be a useful and fun source of information as user will be able to get interesting facts as well as answers to questions in the defined area.

Since the bot's main purpose is to answer questions on these areas, it may not sound very human-like to the user. However, it can response to simple dialogue such as greetings, thank you or even when user talks about themselves.

*Note: the bot will only be able to understand the user if they limit their queries to these topics. In addition, country name must have capital letter at the beginning. Countries should only include sovereign nations, excluding dependencies, this include England, Scotland, Wales and Northern Ireland.

2. Design

2.1. Large scale

2.1.1. Input

Input includes

- Knowledge for the bot to be able to identify keywords, hence understand the user's input and can return an answer if possible. This will be under the form of text file
- User input, in the form of string which may contain keywords for the bot to identify

2.1.2. Output

- Main output: this will be the answer to user's query
- Minor output: this will be a random answer, regarding when user ask for an interesting fact about the bot or just in general

2.1.3 Problems to solve

- Organising, storing and managing knowledge data
- Knowledge data is read and process correctly
- Understand user input, able to identify necessary keywords if they are available
- Look for answers in knowledge if possible, and return the correct answer to the query asked

2.2. Medium scale

The chat bot will be consists of functions which will have knowledge being fed, process and return. The return value will be depending on the purpose of the function and what knowledge has been fed to it.

These functions will be for one of the main purpose - identify keywords and answer.

2.2.1. The query function

This is the function that will achieve one of the main objectives for this project. This function will process user inputs and look for relevant keywords. The bot will again be fed with information and use this along with the user input to return the correct answer.

There should be 3 cases in which user input may fall into:

1. Case: User enters a question with a sufficient amount of keywords
   For example, user can enter "What is the capital city of Vietnam?". The bot should be able to identify certain

keywords, in this case "capital" and "Vietnam. Then it should give a short brief answer "Hanoi".

2. Case: User enters a question with insufficient amount of keywords
   For example, user may enter "How is the weather in Vietnam right now?", the bot can identify "Vietnam", but then it cannot identify any other keywords in that question. In this case it should return "Sorry, I don't have any information on that"
3. Case: User enters a question with no keywords found
   The bot will return "Sorry, I don't know the answer to that".

2.2.2. The returnAnswer function

This function will process exactly two known keywords(which are necessary) in the user input. It will then return the answer as a single word, which should be the correct answer to the user enquiries that contains two keywords used.

These functions will return a random(but appropriate) responses.

2.2.3. The greeting function

This function will have input as a list of general greeting words return a greet to the user when the program start, if user enters a greeting. In the case that the user enters a greeting which is not included in the list of greeting words(for example certain slangs) the bot will answer "Don't you ever say hello?"

2.2.4. The randomFact function

This function will return a random interesting facts about a country/city, when user enters one of the keywords "interesting facts", "fun facts" or "fact of the day".
(There maybe repetition of facts, especially if the user keep asking over and over again as the data fed is limited)

2.2.5. The botFact function

Similar to the randomFact function, but this time the bot will return a fact about itself, completely random when user enters one of the keywords.

2.2.6. The aboutYou function

Some users may enjoy talking about themselves to a chat bot. This function will return a random response when user does, to make the bot sounds more human.

This function will save the user input into a file for pattern analysis(manually done by developer)

2.2.7 The saveInput function

This function will save every user input into a file, from the start till the end of the conversation. It will later be use to analyse patterns of user input in order to improve the chat bot.

2.3. Iterations

Simplest application I will develope will be when the bot is able to response to when user greetings, and to inputs which will have keywords that enables the randomFact and botFact function to be called, and return the appropriate answer. Iterations include:

1. Iteration 1: where all of the three function works with the test
2. Iteration 2: this is where I will start building up the query function. At this stage the bot will be able to store user input in a text file
3. Iteration 3: The bot should be able to return the answer using given keywords
4. Iteration 4: The bot should be able to find a keyword in the user input and return a single answer
5. Iteration 5: The bot should be able to find a keyword and return the answer as a string

In reality, the number of iterations would be unlimited as the chat bot would have to continuously learning from time to time.

2.4. Modules

The program will be splitted into 5 modules, in which 2 will be main and testing(separately) and only one of these two will be used at once, depending on medium or large scale tests, or for users. Each modules, except for main and test will have a header file.

1. Module 1: randomResponse.c. This will contain userGreeting, randomFact, botFact and aboutYou.
2. Module 2: keywordFinding.c. This will contain returnAnswer and userQuery.
3. Module 3: saveInput.c. This will just contain saveInput.
4. Module 4: main.c. This will be the user interface.
5. Module 5: test.c This will be the test for each functions.

3. Test plan

Test for iteration 1 will be test cases for three functions written. For iteration 2, if a keyword is found the function returnAnswer should return the correct answer. For all the other iterations, the function userQuery should be able to return the correct answer in the correct format that each iteration requires.

3.1. Function test(Medium scale)

The functions should be test individually as they are written and finished by a separated test file with different cases. For functions where one or more parameter is passed, there will be at least two test cases to check for validity and error handling.

Functions	Test carried out	Test data	Expected results
userGreetings	Test with correct data	"hello"	Bot returns a random greeting such as "hi" or "hey there" or "hello", etc.
	Test with incorrect data	"ey yo"	"Don't you ever say hello?"
randomFacts	Call the function by assigning to a char pointer	(Function has no parameter passed)	Bot returns a random fact about a country
botFacts	Call the function by assigning to a char pointer		Bot returns a random fact about itself
returnAnswer	Both keywords correct	"capital", "Sri Lanka"	"Colombo & Sri Jayawardenepura Kotte"
		"continent", "South Sudan"	"Africa"
		"continent", "Turkey"	"Asia/Europe"
		"population", "China"	"1,415,045,928"

		"population", "Romania"	"19,580,634"
	One keywords incorrect	"capital", "Hanoi"	"Keyword(s) not found"
	Both keywords incorrect	"what", "hi"	"Keyword(s) not found"
saveInput	Two test cases, each with one string passed to the function	"Line1\n"	Line saved in file inputList.txt, with a blank line after the last line save(which is "Line2\n")
		"Line2\n"	
userQuery	Contain both keywords	"What is the capital of South Africa?"	" The capital of South Africa is Bloemfontein, Cape Town & Pretoria"
	Contain only one keywords	"Where is England located?"	"Sorry, I don't have information on this yet"
	No keywords contained	"What is the weather in Hanoi like?"	"Sorry, I don't have information on this yet"
aboutYou	Call the function by assigning it to a char pointer	(Function has no parameter passed)	Bot returns a random response when user talk about themselves(with the right keywords included)

3.2. UI test(Large scale)

After all of the functions have passed their tests, the user interface will be used to test for everything else. The UI will process user input, look for specific keywords to identify what user is asking for and call upon the functions that will return the correct output. Test for UI include

- The chat bot has been able to identify the necessary keywords.
- All user inputs are saved into a separated file, with the correct format, this requires check if there is a file with such name "inputList.txt" and check that all user inputs are included.
- The chat bot returns the correct output, this is important especially for the userQuery function and the data knowledge for this function.
- Error handling: the chat bot should returns "Sorry, I don't know how to response to this" when user input something unknown

Debugging on a large scale will take place after at least the 4th iteration is complete.

4. Schedule

Work for this project should be done every week. I plan to do a bit of code every day just for about 3 - 4 hours, then spend some time after for revisions and time off.

1. Week 1(20/3 - 25/3): This week will be used for plan and any research related to the project. The large and medium scale plan should be finish by the end of this week.

2. Week 2(26/3 - 1/4): This week will be used to complete the first iteration and test file. The first iteration will also be tested.

3. Week 3(2/4 - 8/4): This week will be used to complete the second and third iterations. I will also start working on the user interface

4. Week 4(9/4 - 15/4): This week will be used to work on the fourth iteration.

5. Week 5(16/4 - 22/4): The user interface should be finish by this week, along with at least the 4th iteration.

6. Week 6(23/4 - 26/4): This final week will be use for completing the final iteration(for this project) if possible, reviewing the code and submission.

How the schedule was followed: I have done the research by the end of first week. The second week was also followed closely with the first iteration complete and tested. However, for week 3 I have had to rethink on the userQuery functions, because it is doing too many things in one function, making it difficult for testing. I have decided to split it up into two separated functions - returnAnswer and userQuery. The algorithm for processing user input isn't as difficult as I thought, and hence I've finished implementing every single functions as well as the user interface in week 4. I spent the last two weeks doing some final touches to the program and finishing up the report.

5. Test outcomes

5.1. User Experience(UX) test

This test will involve letting the user talking to the bot, with just the initial instruction of saying what the bot is about(i.e. it can answer questions on countries).

User inputs will be saved into a file, if the function saveInput has passed all of its tests. It is most likely that user will also try to talk to the bot as a person, and this test is design in order for the developer to see what user will be entering when talking to the bot. The developer then can use this to design the bot, so the bot sounds more human-like, or manipulating with the keywords or algorithms in finding the keywords, so the bot are able to give accurate results with wider range of inputs.

This test would be carried out after all functions are implemented, so that it is then more efficient in terms of building human-like responses since there are real data to analyse.

5.2. Responding speed test

This will check for the amount of time the bot took to read, process and respond to the user input; it will be able to see whether the bot took an amount of time longer than it should to response. Regardless of the size of input, the bot should not take longer than one second to respond. In fact, ideally the bot should take around 0.5 second to read, process and respond.

5.3 Input handling test

This will test how the chat bot interprets when there are more than two keywords. Since the command to return an answer will be execute as soon as the first keyword is found, other keywords, even though known will(and should) be ignored.

For example: for inputs such as "Where is capital population of Latvia" or similar, which doesn't make much sense, the bot should be able to identify "Where" as the first keyword, and it will ignore capital and population, identify Latvia as the second keyword and the output should be "Latvia is located in Europe". Whichever keyword is written first, whether it's a category(capital, where it's located or the population) or a country keyword, the bot will identify it and then execute the command immediately, ignoring all other keywords.

5.4. Accuracy test

This will check whether the data collected as knowledge are accurate. The test will involve looking at the knowledge file, making sure that the spellings for the keywords and answers are correct, all answers match with their appropriate keywords. In

addition, the bot should be able to read both keywords correctly, find the line that contains both and return the appropriate answer, in the appropriate format without any errors.

6. Reflections

By completing this project, I have learn the basic idea of an artificial intelligence, in terms of how it process user input in order to understand it and return the appropriate answer. I am now able to organise the knowledge file for the chat bot to process along with the user input, at a basic level. In addition, I have also learn and understand many string operations of C programming.

My strength would be my efficiency. This is proven by the fact that I have finished the entire project(according to the plan) earlier than the schedule. I feel like I'm better at programming and at C now, since I have done this project all by myself during the holiday and I'm confident that I can tackle problems at this level of challenging.

My weaknesses are mostly in the planning stage. I did not state this explicitly in my schedule, but I have made a plan before this. However, when I try to write out the algorithms and code one of the functions, I ended up finding that it has very high complexity and that I wouldn't be able to understand the code that I need for that.

With the knowledge I have gained by completing this project, I will be able to use it in further studies throughout my course. This project is a good start for me since I'm going to study Computer Science with Artificial Intelligence, although Artificial Intelligence is a compulsory module for every Computer Science degree, having deeper knowledge can become more useful when it comes to the group project at the end of the year.