Chat Bot using NLP - NLP ChatBot

Btech Cse (Chandigarh University)

Scan to open on Studocu

Studocu is not sponsored or endorsed by any college or university
Downloaded by HASSAN ALI (hxn112233@gmail.com)

CHATBOT USING NLP

ABSTRACT

The increasing use of technology is changing the way students learn and absorb information. Hence, educational institutions also need to accelerate their student communication process to draw the attention of this fast-paced generation. The project here aims to harvest better teacher-student relationships and to prepare students for personalised learning by

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

developing a convincing chatbot to assist pupils in their queries related to common operations in numerical base 2 or binary operations with general information operations. Background research on various web applications and chatbot systems was discussed and analysed before the development of the project. Functional and non-functional requirements were assessed on the priority basis and the respective methodology was devised for the project. Python Flask and MongoDB are used for the backend whereas HTML5, CSS3 and jQuery are used for the frontend of the framework. Deployment is done through Heroku Cloud Platform. Moreover, MongoDB Atlas, NLTK, AWS Lambda Functions, AWS Lex, Boto3, Pymongo and Gunicorn are used as Third-Party Tools and Libraries. Various test cases were verified to check for the functional and non-functional requirements completion. A Likert-scale questionnaire survey was done from 60 engineering students who have studied Introduction to Digital Electronics module after the chatbot deployment and the survey responses were recorded and evaluation of the results were generated. The survey concludes that majority of the participants find the chatbot to be extremely useful and the developed framework generates correct responses, and explanations to the questions. Further to the aforementioned points, future study for the project has been proposed in the paper. AI approach can be used to train the chatbot framework and to provide more accurate answers for students in future. Moreover, the framework can be altered to support advanced parsing techniques in future along with an added voice functionality and multi-language support in the chatbot framework.

Downloaded by HASSAN ALI (hxn112233@gmail.com)

DECLARATION OF ORIGINALITY

I declare that the research described in this thesis is the original work of the author except where otherwise specified, or where an acknowledgement is made by reference.

The work was carried out at the School of Engineering, London South Bank University. Under the supervision of Dr. Oswaldo Cadenas.

This work has not been submitted for any other degree or award at any other academic or professional institution during the research program.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

ACKNOWLEDGEMENTS

The author would like to thank ________________ for his encouragement, guidance and support throughout the research programme, from its original inception through to completion.

The ideas and suggestions contributed by the staff at CU, and by other professionals in the computer science and chatbot framework development sector are also gratefully acknowledged.

Finally, I would like to thank

Downloaded by HASSAN ALI (hxn112233@gmail.com)

TABLE OF CONTENTS

ABSTRACT....1
DECLARATION OF ORIGINALITY....3
ACKNOWLEDGEMENTS....4
LIST OF FIGURES....8
LIST OF TABLES....9
LIST OF ABBREVIATIONS....9
CHAPTER 1....10
INTRODUCTION....10
  Overview....10
  Scope of the Project....13
    Need of Chatbots....13
  Research Objectives....15
    Qualitative Data....15
    Personalised Learning....15
    Establishing a stronger teacher-student relationship....15
    Anytime assistance....16
    Customised Teaching....16
  Research Outline....16
CHAPTER 2....17
LITERATURE REVIEW....17
  Earlier Chatbots....18
  Self-learning Chatbots....20
  Existing models in education sector....21
CHAPTER 3....27
METHODOLOGY....27
  Technology Stack and Web Applications....28
    Backend....28
    Frontend....28
    Deployment....29
    Third Party Tools and Libraries....29
  Functional Requirements....31
    API Calls....31
    Generic Question Construction....31
    Generic Answer Construction....31
    Query Handling....32
    Information Management....33
  Non-Functional Requirements....33

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Accuracy....33
Fast Response....33
Portability....33
Maintainability....34
Ethics....34
Answering Generic Questions/Bot Conversion....34
Binary Module....34
Data Storage....36
CHAPTER 4....37
FINDINGS AND RESULTS....37
Overview....37
Binary Module....37
Test Case 01: Chatbot’s response for Addition of Binary numbers....37
Test Case 02: Chatbot’s response for Subtraction of Binary numbers....38
Test Case 03: Chatbot’s response for Addition of Decimal numbers....39
Test Case 04: Chatbot’s response for Subtraction of Decimal numbers....40
Test Case 05: Chatbot’s response for decimal to binary conversion....42
Test Case 06: Chatbot’s response for binary to decimal conversion....43
Test Case 07: Chatbot’s response for unsigned decimal number’s one’s complement....44
Test Case 08: Chatbot’s response for signed decimal number’s one’s complement....45
Test Case 09: Chatbot’s response for unsigned binary number’s one’s complement....46
Test Case 10: Chatbot’s response for signed binary number’s one’s complement....47
Test Case 11: Chatbot’s response for unsigned decimal number’s two’s complement....49
Test Case 12: Chatbot’s response for signed decimal number’s two’s complement....50
Test Case 13: Chatbot’s response for unsigned binary number’s two’s complement....52
Test Case 14: Chatbot’s response for signed binary number’s two’s complement....53
Test Case 15: Chatbot’s response for unsigned decimal’s binary representation bits requirement....54
Test Case 16: Chatbot’s response for signed decimal’s binary representation bits requirement....55
Test Case 17: User Input Bits Functionality....56
General information about the university....65
Generic Questions/Answers....67

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Evaluation of Results......69
Evaluation on the basis of question type......72
Evaluation by comparison with existing models......73
CHAPTER 5......76
DISCUSSION......76
Research Objectives Analysis......76
Pros & Cons of the implementation of Chatbot framework......77
Pros......77
Cons......77
Questionnaire Survey and Results......78
Responses Recorded......78
Survey Analysis......80
CHAPTER 6......82
CONCLUSION......82
Summary......82
CHAPTER 7......84
LIMITATIONS & RECOMMENDATIONS......84
Framework Limitations......84
Text-based Input......84
Rule-based approach......84
Single Language Support......84
Framework Testing Limitations......84
Future Work......85
Machine Learning......85
Voice-based......85
Advanced NLP......85
Better Interface......86
Multiple Language Support......86
CHAPTER 8......87
RESEARCH ETHICS......87
APPENDIX A. Anatomy of a self-learning chatbot......88
APPENDIX B. Questionnaire Survey......89
APPENDIX C. NLP......90
Tokenization......90
Function 01: Input String Format Checking......91
Function 02: B2D Conversion......92
Function 03: One’s Complement of a Signed Integer......92
Function 04: Two’s Complement of a Signed Integer......94
Function 05: D2B Conversion......97

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Function 06: Signed Decimal to Binary Conversion..98
Function 07: One’s Complement of an Unsigned Integer..................................100
Function 08: Two’s Complement of an Unsigned Integer.................................101
Function 09: Bit Representation..105
Function 10: Two’s Complement of a Binary Integer..106
Function 11: Two’s Complement of a Decimal Number....................................107
Function 12: Get Decimal String..108
Function 13: Addition/Subtraction..108
Function 14: Controller...116
APPENDIX D. SERVER..119
APPENDIX E. LEX...121
APPENDIX F. Atlas Database..122
Create Mongo dB Instances...122
Adding data to Mongo database..123
REFERENCES...123

Downloaded by HASSAN ALI (hxn112233@gmail.com)

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

JSON	JavaScript Object Notation
CSS	Cascading Style Sheet

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

FAQ	Frequently Asked Questions
Q&A	Questions and Answers
AIML	Artificial Intelligence Markup Language
XML	Extensible Markup Language
MSN	Microsoft Network
NLP	Natural Language Processing
AI	Artificial Intelligence
API	Application Programming Interface
BSON	Binary JavaScript Object Notation
AJAX	Asynchronous JavaScript and XML
PaaS	Platform as a service
DBaaS	Database as a service
NLTK	Natural Language Toolkit
AWS	Amazon Web Services
HTML	Hypertext Markup Language

CHAPTER 1

INTRODUCTION

Overview

A chatbot is a term refers to a computer program, with the ability to communicate with people via auditory or textual methods. Using different techniques of Natural Language Processing (NLP) in such a way that it somehow replicates the human way of conversation and provides a better way for people to get information from the computer [1]. Chatbots are also

Downloaded by HASSAN ALI (hxn112233@gmail.com)

called conversational agents in a formal way for literary scientific work. Natural Language Processing (NLP) is used in chatbots to generate humans like language, by using different algorithms on natural language rules to make it understandable for computer [2]. If the user provides voice input, it is first converted to text and then the text is processed through NLP algorithms. Which extract information from each sentence, converting it into a machine-understandable form. NLP doesn’t guarantee 100% results since it has its own limitations. Because developers are still quite far away in being able to replicate how humans work and how they communicate. So, in some cases it may lead to unexpected results.

One of the main techniques used in NLP is Syntactic Analysis. In syntactic analysis, we analyse the syntax of the sentence [3]. Syntax is the rules of any language which provides a way of a meaningful conversation by following these sets of rules. NLP uses the syntax of the language in which the input is given, parse that language by following those set of rules, and extract the intended information for the machine to process it. After getting the required information, chatbot performs the desired task, and in the end provides a response which is understandable for humans as well [3]. Major techniques of syntactic analysis include lemmatization i.e. reduction of adjusted form to single forms. Morphological segmentation i.e. morphemes are built from individual units. Word segmentation i.e. distinct units are built from the continuous text. Part-of-speech tagging i.e. identification of parts of speech. Finally parsing i.e. input string is grammatically analysed [3].

Parsing technique[1] is used to achieve the natural language processing in this project.

---

1 See Chapter 3

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

NLP tasks can be approached either by Rule-Based or Self Learning[2] [4]. There are a fixed set of rules defined for NLP algorithms to parse user input, which varies in its complexity in Rule-based approach. It focuses on pattern-matching or parsing of user input centred on the stipulated set of rules, which fills the gaps between the unstructured sentences to extract its meaning [4]. They are mostly used to handle a specific task with limited scope and set of rules, and it can be highly efficient in such scenarios providing almost 100% accuracy. Whereas, in Self Learning approach, we use machine learning algorithms for building the bot. For this model, we need to provide a huge amount of data to the framework. The data is first provided to the training model, so it can provide answers based on what it learnt from previous data [5]. There are multiple ways to generate data for learning, and the best possible approach is to first create a bot on the established of rule-based approach. Then apply machine learning algorithms based on the data collected.

There are two basic models for self-learning approach i.e. Retrieval Based or Generative. In Retrieval Based model predefined responses are previously stored in the database, and when a specific question is asked it retrieves the best response to answer the question effectively. The pre-defined responses can come from various sources, it can be data gathered from previously asked questions or any other manually saved variables [6]. Though in the generative model, the chatbot parse the user input word by word and generate the answer based on the closest match. Which makes it able to generate new replies instead of existing ones. This gives the aura of intelligence to present unique responses based on the user's input. It is

---

[2] See Appendix A

Downloaded by HASSAN ALI (hxn112233@gmail.com)

more complex than other models and requires more training data to increase efficiency [6].

Rule-based NLP approach is used to develop the chatbot framework in this project.

Scope of the Project

Need of Chatbots in Educational Institutions

An exceptional and well-reputed educational organisation is not only one with profoundly competent teachers, thoroughly equipped facilities and internally certified courses. But the one that offers sufficient support to their students. Thus, it is mandatory for an institution to always guide their students exceedingly, by giving them up to date and solid knowledge. However, it is nearly impossible to make sure each student is being taken care of fully without any failures. A chatbot can become in part a teacher's aide rather or a simple lesson tool, and as a result students' will naturally be more engaged [7]. With the availability of a chatbot, students would not be distracted while waiting for the teacher to attend to them.

It is common nowadays that students often put 'do my assignment requests' over the internet in the hope that they can find someone to aid them in the completion of the assignments. Or look for prompt answers for precision regarding the courses they are taking. Similarly, teachers also require some time-saving alternatives to their repetitive processes that undergo all throughout the year. The institution can set-up simple answers for their chatbot, and there will be no need to manually search for them. As for the teachers they can also take advantage of a chatbot in several ways, to facilitate both their lecturing and assessment. Even though any developed

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

framework will be for a specific university or for a specific module only. But still, it will appeal to a large number of people.

In everyday life, the rising use of modern technology is altering the way students acquire and grasp information. The way we teach the next generation has always changed with the addition of technology. Technological devices are continually flourishing and more getting advanced, and new generations are getting involved with them like never before. Therefore, to draw the attention of this fast-paced generation, the educational institutions also need to enhance their student communication links [8].

When it comes to promotion of chatbots in educational processes, Shawar and Atwell state that it is mandatory to remember that the teacher is the backbone of the teaching process, and the learning technology may act as a booster but not a restoration [9]. Let’s take an example, when a chatbot is used to answer the questions of students. The teacher can use set-up log files of the discourse to see which students are having problems and what their shortcomings are. Thus, the teacher can use a chatbot to look for quandaries as the students use it to solve them.

Continuing with the predicament introduced earlier that chatbots are still not commonly released in the education sector. The aim of this thesis is set this in context, for the more the advanced evolution of educational chatbots. It is important to address what has been done, and to summarise this knowledge in a concise plus relevant manner. Therefore, the research objectives of this study is to document the instructional uses and aptitudes of a chatbot, in an educational context by reviewing the literature in the field

Downloaded by HASSAN ALI (hxn112233@gmail.com)

i.e. looking at what the chatbot can do, and be used for in an educational context.

Research Objectives

Qualitative Data

Teachers only hear and interpret the questions of the most vocals' students. They are not aware of many of the queries and encounters faced by the majority of their students. Educational Chatbots can change this. Teachers would be capable to see the chat history and point out the areas that they may want to intensify more in class. In addition to this, they can view questions that the chatbot was unable to answer adequately. Thereafter the teacher can reach out to students individually to provide the proper approach to solve the questions.

Personalised Learning

Every student has different skills, interests, and capabilities. So personal tutors such as a chatbot can provide one-on-one lectures to the students. Unfortunately, even some of the most highly certified and expensive educational institutes in the world, are not able to provide these kinds of services [9].

Establishing a stronger teacher-student relationship

In present-day memorising lessons and tests grades, are entirely the main focal point of most of the teachers. By taking charge of these tasks, chatbots will help teachers to fixate on cultivating an exclusive relationship with their students. They will have the favourable circumstances to provide them with personal guidance and elaborate the curriculum with their own research perceptions. In this way, teachers can also be able to provide

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

distinctive mentorship. As a consequence, this will be notably beneficial for students with learning disabilities.

Anytime assistance

With educational chatbots at work students, would not have to wait anymore to have their queries attend to and acquire the relevant knowledge they are counting on.

Customised Teaching

Chatbots can grant assistance to teachers who are not able to give consideration and attention to the needy students individually. Contemporarily, electronic gadgets such as computers, laptops, and other devices are becoming more intensely used as learning tools. Chatbots are being installed that can tag along with students whenever they are in lectures and doing their assignments. Thus, the chatbots prevent the students from falling behind their peers.

Research Outline

Chapter 2 covers the background literature research, which provides an overview of chatbots applied in education sectors worldwide, chatbot framework developed through the years, existing frameworks, self-learning chatbots, interactions of bots with humans and tech stack used in the framework development. Chapter 3 covers Methodology which includes the functional and non-functional requirements, generic questions/answers module, binary module and data storage. Chapter 4 details the Findings and Results. It includes the various test cases for the binary module and generic questions/answers module. Evaluation summary of the test results is tabulated in this chapter. Chapter 5 recounts the discussion and analysis of

Downloaded by HASSAN ALI (hxn112233@gmail.com)

the framework. It includes the questionnaire survey results and the summary analysis of the survey. Chapter 6 contains the conclusion and recommendations. It consists of a summary of the research project and future work.

CHAPTER 2

LITERATURE REVIEW

The core principle of a chatbot, is that it will work with humans perfectly and follow their commands and needs. A test to check the performance of a bot was conducted by Alan Turing in 1950 [10] where a human judge would have to predict if the other communicating entity was a computer machine or a person. The test was very fruitful and is still notably efficacious today. The chatbots hold substantial drawbacks, namely they cannot book a table at the restaurant or order the food on their own. They will do as they were previously instructed to do, nonetheless they are incapable of thinking on their own. General conjecture chatbot designers keep in mind, is that users typically have an intent they want to achieve by the end of the conversation when they initiate an interaction with a chatbot. This then influences the conversation’s flow and topics, in order to reach the chosen goal. This can be exploited by developers since certain patterns of behaviour tend to arise as a result.

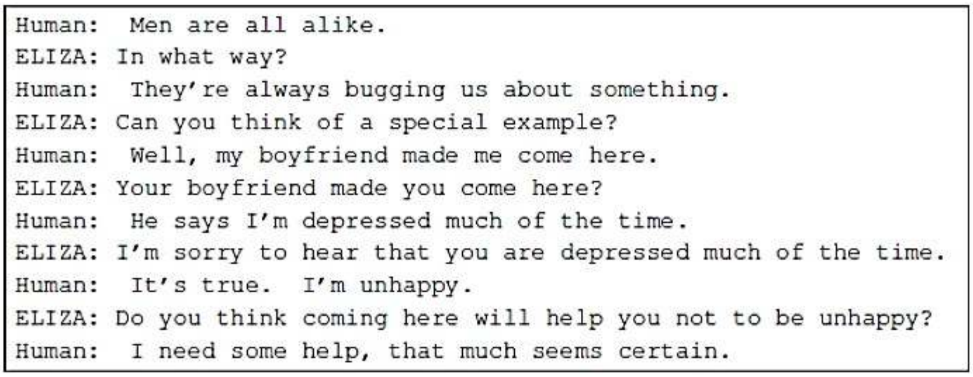
This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Earlier Chatbots

The first conversational framework was a PC program invented in 1966 by Joseph Weizenbaum named as ELIZA. ELIZA’s framework was comprised of basic NLP and keyword matching technique [11]. ELIZA matches the keyword defined in its NLP and gave the answer. In the case where there was a no keyword match, then ELIZA would continue the conversation by asking further questions about the user. To illustrate, if a user asked ELIZA how he/she can contact his/her mother. Then the keyword ’mother’ will be matched which will be stored in ELIZA’s NLP. It will next generate some standard replies and will be asking more questions about the user’s family.

The constraint with the framework of ELIZA was intelligence. There was no intelligent conversation between the user and the ELIZA. The system simply tried to prolong the conversation, so that user would be encouraged to ask auxiliary questions. When no keywords were matched, ELIZA would bestow conventional responses stored in its NLP by its creator e.g. “Very interesting, please go on” or “Can you think of a special example?”



```
Human:  Men are all alike.
ELIZA: In what way?
Human:  They’re always bugging us about something.
ELIZA: Can you think of a special example?
Human:  Well, my boyfriend made me come here.
ELIZA: Your boyfriend made you come here?
Human:  He says I’m depressed much of the time.
ELIZA: I’m sorry to hear that you are depressed much of the time.
Human:  It’s true.  I’m unhappy.
ELIZA: Do you think coming here will help you not to be unhappy?
Human:  I need some help, that much seems certain.
```

Figure 2.1: ELIZA’s response

Source: Adapted from [11]

After ELIZA, in 1995, another conversational framework was constructed by Richard Wallace and he named it as ALICE. ALICE used advanced NLP i.e. pattern matching technique rather than keyword-matching technique, and

Downloaded by HASSAN ALI (hxn112233@gmail.com)

stored the information in AIML files [12]. ALICE was able to use any of the AIML categories from atomic, default and recursive categories.

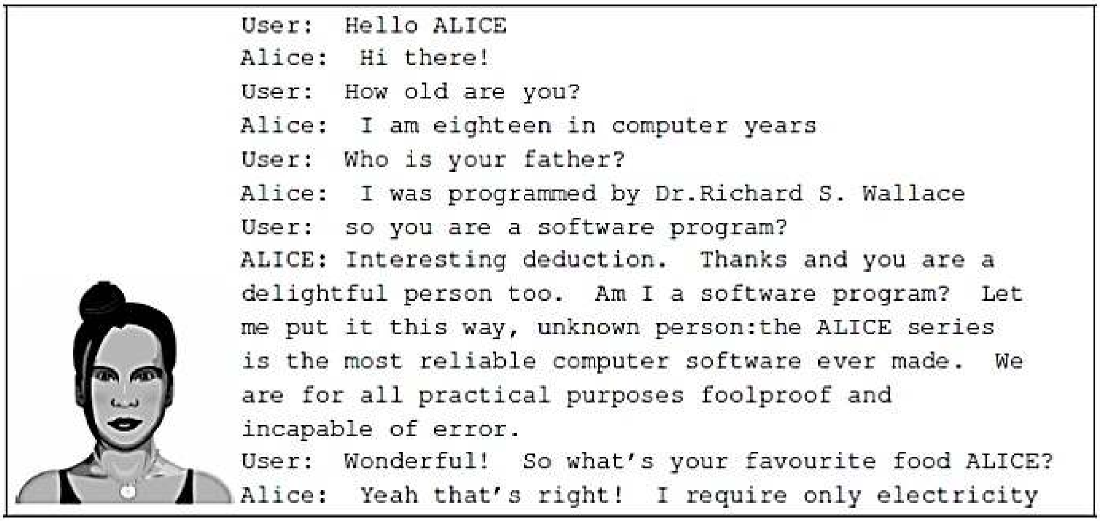
If ALICE employed atomic categories, it means that the exact pattern of the input string should be matched. For example, in response to the question “What is your name?”, the framework will match the exact pattern. Therewith if the pattern is successfully matched, it will yield an answer to the asked question. That being said, If ALICE ran default categories that indicate that a wild character * will be used to match the input string. Therefore in response to the question, “What is *”, the structure will match the wild character with the input string. Provided the wild character is appropriately matched, it will supply an answer to the asked question. Where ALICE was not able to offer the user with an answer with the first category. It will jump to the second category and will again check for the wild character match. Finally in the case where ALICE operates with recursive categories. Then ALICE will use special tags to recurse, so that the user can ask questions more specifically. Recursive categories engage wild character as well but do this to reduce the submitted input string. For example, if the user asks “Tell me what is the date today”. After recursing of special tags and using a wild character to reduce the input string. The question will simply become “What is the time?”.

By the same token, punctuations from the sentences are removed, and the filtered sentence is passed though the recursive categories to find the longest match in the sentence. For example, if a user asks, “Tell me the date using Year, Month and date”. Then it would be filtered out to be, “Tell me the date using Year Month and date”.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Likewise to ELIZA, the shortcoming with the framework of ALICE is also intelligence. The system is viable as long as the data is stored in the database. Hence, both the ELIZA and ALICE do not have the faculty to learn.



```
User:  Hello ALICE
Alice:  Hi there!
User:  How old are you?
Alice:  I am eighteen in computer years
User:  Who is your father?
Alice:  I was programmed by Dr.Richard S. Wallace
User:  so you are a software program?
ALICE: Interesting deduction.  Thanks and you are a
delightful person too.  Am I a software program?  Let
me put it this way, unknown person:the ALICE series
is the most reliable computer software ever made.  We
are for all practical purposes foolproof and
incapable of error.
User:  Wonderful!  So what's your favourite food ALICE?
Alice:  Yeah that's right!  I require only electricity
```

Figure 2.2: ALICE's response

Source: Adapted from [12]

Following this another intelligent framework was built by ActiveBuddy, Inc. in 2001, and was named SmarterChild. The key enhancement was that a learning database was linked with the framework, to help train the model and be resourceful for its clients in future [13].

Self-learning Chatbots

Learning chatbots have the forte to learn from what the client tells it, or the data the client inputs. The guidelines from the operator are very important and helpful in this regard. There are two rudimentary types of ways to learn. The first is the chatbot will do what the user will ask it to do. It will record the saying or the instructions, and follow back accordingly what is inputted. The database will then store the answer to the question, and will answer back with the text afterwards. The data whether it be valid or mistaken, will also

Downloaded by HASSAN ALI (hxn112233@gmail.com)

be stored to respond to other users. The expression or commands told to the chatbots are extremely pivotal for its better understanding and working with human beings or the living environment. The other is that all answers will be saved and the most related answer will be used to be replied back.

Momentous headway was made for the conversational specialist community by a group at IBM. Through the Watson AI venture that had been fostered since 2006. The system was conceived with the sole reason of winning the American TV show Jeopardy [14]. This was intriguing from an NLP perspective, because the inquiries on the show include a great number of figures of speech. Plus require quick data recovery from large information bases. Alas, this AI in its past structure could just respond to joke questions and was not able to carry on an appropriate discussion with another person [14].

In the mid-2010s came the rise of remote helpers, for example, Apple's Siri, Microsoft's Cortana's, Google collaborator, Amazon's Alexa and others.

An additional prominent event in the field of chatbots was the arrival of the Messenger Platform for Facebook Messenger in 2016, This led to the formation of conversational operators for non-AI related arrangements.

Existing models in education sector

The application of chatbots has rapidly increased in the educational sector lately. Academic institutions are more prone to state-of-the-art and productive ways of putting chatbots into effect as compared to past in their usual routine practices.

Staffordshire University from the United Kingdom has developed Beacon[3], where thousands of students within the institution can search for the

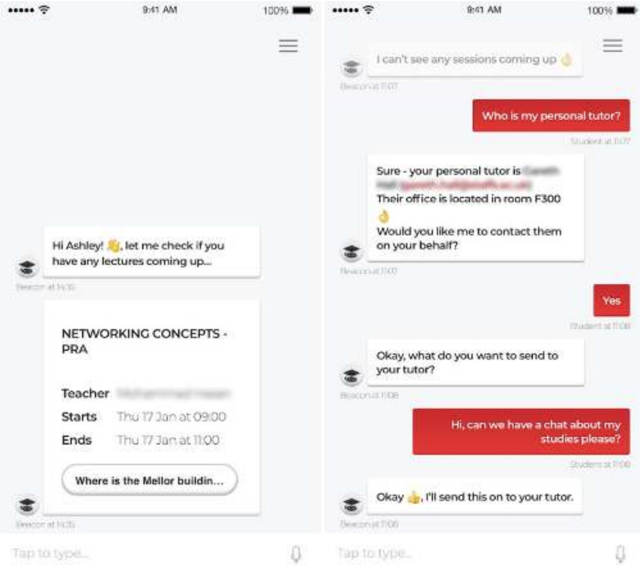
---

[3] Chatbot Application

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

instructors for their courses. Pupils can also inquire about their lectures schedule and location using their voice or by typing. They can also attain the answers to 400 most frequently asked queries, covering topics such as facilities and support services [15] as shown in Figure 1 below.



Hi Ashley! 👋, let me check if you have any lectures coming up...

NETWORKING CONCEPTS - PRA

Teacher [redacted]
Starts Thu 17 Jan at 09:00
Ends Thu 17 Jan at 11:00

Where is the Mellor buildin...

I can't see any sessions coming up 👌

Who is my personal tutor?

Sure - your personal tutor is [redacted]
Their office is located in room F300 👌
Would you like me to contact them on your behalf?

Yes

Okay, what do you want to send to your tutor?

Hi, can we have a chat about my studies please?

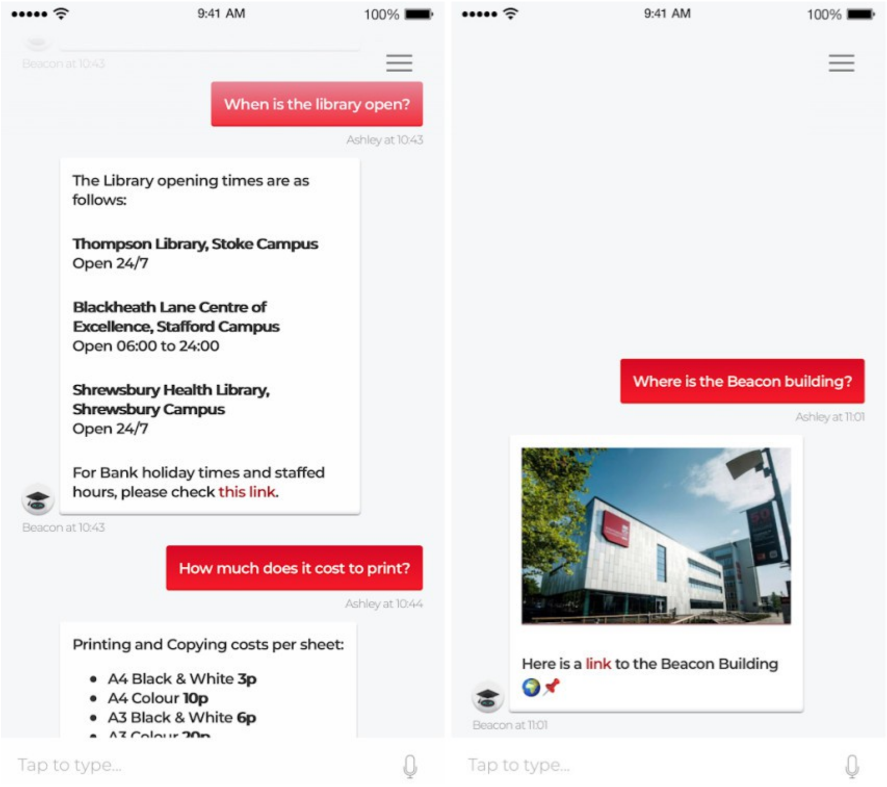
Okay 👍, I'll send this on to your tutor.

Figure 2.3: Beacon by Staffordshire University

Source: Adapted from [15]

Students can also request other general queries e.g. location of specific lecture halls, library and soon. The opening time of the library or cafeteria, printing and copying costs for various sizes of papers et cetera. As shown in Figure 2 below.

Downloaded by HASSAN ALI (hxn112233@gmail.com)



When is the library open?

Ashley at 10:43

The Library opening times are as follows:

Thompson Library, Stoke Campus
Open 24/7

Blackheath Lane Centre of Excellence, Stafford Campus
Open 06:00 to 24:00

Shrewsbury Health Library, Shrewsbury Campus
Open 24/7

For Bank holiday times and staffed hours, please check this link.

Beacon at 10:43

How much does it cost to print?

Ashley at 10:44

Printing and Copying costs per sheet:

- A4 Black & White **3p**
- A4 Colour **10p**
- A3 Black & White **6p**
- A3 Colour **20p**

Where is the Beacon building?

Ashley at 11:01

Here is a link to the Beacon Building 🌍📌

Beacon at 11:01

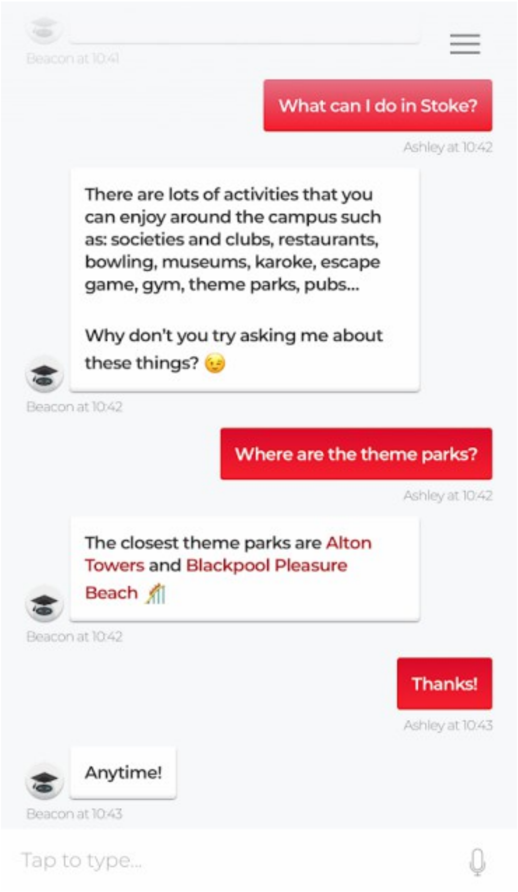
Figure 2.4: Requesting Location from Beacon by Staffordshire University

Source: Adapted from [15]

Beacon was created in combination with ANS Group a cloud service provider, and hosted on the forum Microsoft's Azure stage. Staffordshire University believes that Beacon will improve with time as further students will use it, and will also help to solve their non-academic issues [15]. For example, students from the fresh intake can use Beacon to ask for any nearby outing places e.g. parks, museums, restaurants and so forth. Compatibility people not belonging to Staffordshire University have to rely on a search engine to find such information [15] as shown in Figure 3 below.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



Beacon at 10:41

What can I do in Stoke?

Ashley at 10:42

There are lots of activities that you can enjoy around the campus such as: societies and clubs, restaurants, bowling, museums, karoke, escape game, gym, theme parks, pubs...

Why don't you try asking me about these things? 😉

Beacon at 10:42

Where are the theme parks?

Ashley at 10:42

The closest theme parks are Alton Towers and Blackpool Pleasure Beach 🎢

Beacon at 10:42

Thanks!

Ashley at 10:43

Anytime!

Beacon at 10:43

Figure 2.5: Beacon: Requesting non-academic information from Beacon by Staffordshire University

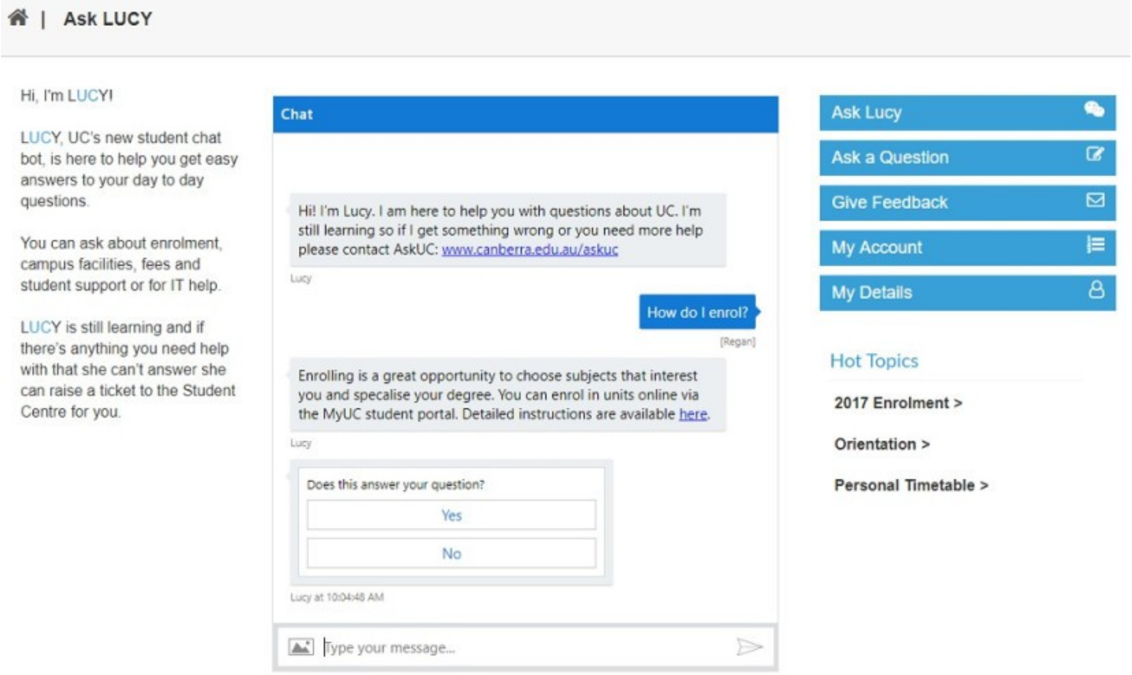
Source: Adapted from [15]

The Staffordshire university intends to include more features in the framework in the near future. For instance, if a student needs to set a reminder for a specific class, or someone needs a recommendation for a book or journal [15]. Andrew Proctor, who is an executive of advanced administrations at the Staffordshire University said that, " Over time, we expect that students will have more daily interactions with Beacon than anyone else at the university. So it will be one of our most important tools" [15]. Andrew also added that, "Going to university can be stressful and is often the first time a teenager will move out of their home to live somewhere new. Beacon is there to help them; it's not just a Q&A bot. In Welcome Week, it can recommend societies that will help them make friends, and it

Downloaded by HASSAN ALI (hxn112233@gmail.com)

will eventually guide new students toward services. If they need extra support during the first few months. It will ask them how their lectures are going to ensure that if they are struggling, we can help them quickly and in the best way" [15].

The University of Canberra devised various digital assistant frameworks including Bruce and Lucy to help both students and faculty in 2017 [15]. Bruce was designed to be a faculty-centred chatbot, whilst the purpose of Lucy was to be a student-centred chatbot. The responsibilities of Bruce include, going through staff questions and responding to them. Inquiring if they have received their paycheck, available off-days etc. Whereas, Lucy handles the queries related to student enrollment, student orientation, timetables and suchlike [15] as shown in Figure 4 below.



Hi, I'm LUCY!

LUCY, UC's new student chat bot, is here to help you get easy answers to your day to day questions.

You can ask about enrolment, campus facilities, fees and student support or for IT help.

LUCY is still learning and if there's anything you need help with that she can't answer she can raise a ticket to the Student Centre for you.

Hi! I'm Lucy. I am here to help you with questions about UC. I'm still learning so if I get something wrong or you need more help please contact AskUC: www.canberra.edu.au/askuc

Lucy

How do I enrol?

[Regan]

Enrolling is a great opportunity to choose subjects that interest you and specalise your degree. You can enrol in units online via the MyUC student portal. Detailed instructions are available here.

Lucy

Does this answer your question?

Lucy at 10:04:48 AM

Figure 2.6: Lucy by University of Canberra

Source: Adapted from [15]

Additionally they are planning to commence more chatbot frameworks to calculate the merit points for the students by analysing their ATAR score.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Subsequently ranking the students against the programs offered and programs they eligible for based on their ATAR merit points [15].

Arizona State University and the University of Memphis deployed AdmitHub. To achieve advancement in the Artificial Intelligence of the chatbot framework [15]. The produced framework has been successfully running, and within the first month after its launch in excess of 70% new students were engaged by the chatbot [15]. Furthermore the chatbot is so successful at answering questions accurately, that 99% of the asked queries were remedied and the university’s administration was saved from the hectic process [15].

A common problem every university often face that graduates often fail to register in the fall after successfully gaining admission. Hence to counter this problem, a chatbot was dispersed by Georgia State University called Pounce. To engross students and assure they matriculated as planned [15]. In the first summer of running, Pounce delivered more than 200,000 answers to questions asked by incoming freshmen [16]. This translated into an additional 324 students sitting in their seats for the first day of classes at Georgia State, rather than sitting out the college experience [16].

Georgia Tech is another college in the Peach State that has been utilising chatbots. An educator of software engineering has harnessed the innovation to fill in as one of his nine instructing associates. The TA chatbot responds to students’ matters by posting important notices and due dates online. The chatbot is tailored to talk normally, have exclamation points usage and respond positively with, "That's right." The chatbot can tackle 40% of the inquiries posted. As a result, this enables different TAs to manage more and more high-level work [17].

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Moreover, Universidad Siglo 21 has put into practice a virtual cognitive assistant named AgentBot, which supports students with day-to-day conundrums about the university and life on the campus. The service also aids students with their enquiries about the course content. The solution uses AIVO's agent bot [18].

Deakin University's Genie App is a smart personal assistant that too services students during their course study. The ability to connect to various campus datasets amplifies Genie's aptitude to provide on-demand support to all students on the campus. The app also bolsters speech-to-text and text-to-speech interaction [18].

CHAPTER 3

METHODOLOGY

Rule-based NLP approach will be devised to develop the chatbot framework. The sole responsibility of the developed framework is to deliver a user-friendly interface. That users can access the online cloud platform and procure a solution to their enquiries. The prepared framework upholds

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

generic questions about the university, courses information etc. along with the queries correlated to binary operations.

Technology Stack and Web Applications

Following technology stack and web applications are used to develop the chatbot framework.

Backend

Flask

Flask is a Python based open source micro-framework foremost designed for REST APIs (Application Programming Interfaces). It’s licensed under BSD Unlike other frameworks such as Django or .NET which are considered batteries includes, Flask remains lightweight by only offering handful yet core set libraries. These include Werzeug (local server), Jinja (templating engine for web pages), XML and JSON parsers along with several others. Because of this, Flask’s learning curve is relatively flat and allows greater customisation, in terms of what sort of application structure does a developer want.

MongoDB

An open-source NoSQL database implemented in C++. Unlike traditional Relational Databases such as MySQL/PostgreSQL, MongoDB doesn’t require a rigid schema. Its flexible nature allows developers to store data in documents, nested or sequential both.

Frontend

HTML5

Hyper Text Markup Language for displaying web pages on the browsers. HTML5 is the latest iteration of the standard which adds new tags including navigation bars, header, footer, article and audio files.

Downloaded by HASSAN ALI (hxn112233@gmail.com)

CSS3

Cascadian Style Sheets or CSS is a language that describes the styling of the content being displayed inside web pages. This includes, font styling, content alignment, animations, grid and tabular layouts etc. CSS3 is the latest version which incorporates responsive designs features through media queries and flex box.

jQuery

Javascript library for developing dynamic frontend for web applications. It allows the handler to asynchronously exchange data with the server using AJAX (Asynchronous Javascript and XML).

Deployment

Heroku

Platform as a Service which allows developers to easily deploy, manage and scale their web applications on cloud. Heroku basically utilizes Dynos which is the lightweight app-container, virtualised instances of Linux machines. Heroku supports the application of multiple frameworks including Flask, Django and Ruby on Rails. Their most basic (and free) model offers up to the launch of 5 projects, each having PostgreSQL database and single Dyno.

Third Party Tools and Libraries

MongoDB Atlas

DataBase as a Service offered by Atlas. The reason for choosing this service was that in the initial stage MongoDB instance was selected to run on Heroku, but payment was required. The only alternative option which was indistinguishable in quality, easy and free of cost to use was Atlas.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

NLTK

Natural Language Toolkit or NLTK is a Python founded library. Centrally formed for resolving Natural Language Processing concerned problems. Its chief attributes include string parsing, tokenization, 50 different corpuses, text classification, sentiment analysis etc.

AWS Lambda Functions

Is a computing platform which is often regarded as server-less, since they let you run the code without the server management.

AWS Lex

Amazon offers Lex as part of its AWS suite for building conversational interfaces for all sorts of applications. It purveys a formidable NLP interface in order to analyse users' questions and call upon the desired intent. Intents are used in Lex to generate a specific response. Amazon Lex intent triggers AWS Lambda function, it takes the question and passes it to a search engine (Wolfram Alpha), in our case via an API call. Lastly transferring the JSON response back to our application server. Like Heroku and Atlas, both Lex and Lambda functions have their respective free tiers for small scale usage or experimentation.

Boto3

Boto3 is a Python client for using all sorts of AWS (Amazon Web Service) linked offerings.

Pymongo

MongoDB client for Python.

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Gunicorn

Application server for backend similar to Werzeug. It was elected given Heroku Dynos prohibits the support of Werzeug. Accordant with official documentation only Gunicorn is endorsed.

Functional Requirements

API Calls

1. Parsing of the input string will be done from the flask server **Priority 1**
2. JSON response documents will receive api data from the flask server. **Priority 1**

Generic Question Construction

1. URL will send text string to the module. **Priority 1**
2. Keywords defined in the NLP will be identified from the input string and replace them with generic representations. **Priority 1**
3. An output string will be generated. **Priority 1**
4. Keywords defined in the NLP will be stored in the module e.g. complement, bits, convert etc. **Priority 1**
5. An error message will be generated in case of null parsing, “Sorry, can you please repeat that?” **Priority 1**

Generic Answer Construction

1. Parsed output string will be sent as an input to this module. **Priority 1**
2. An appropriate answer will be generated from this module after keyword matching from the NLP e.g. “Who are you?” becomes “Hi I'm Serina.” **Priority 1**
3. An error message will be generated in case of null keyword matching, “Sorry I don’t know that?” **Priority 1**

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Query Handling

1. Students should be permitted to ask for information about the university by the framework. **Priority 1**
2. Students should be permitted to ask for queries related to binary addition by the framework. **Priority 1**
3. Students should be permitted to ask for queries related to binary subtraction by the framework. **Priority 1**
4. Students should be permitted to ask for queries related to decimal addition by the framework. **Priority 1**
5. Students should be permitted to ask for queries related to decimal subtraction by the framework. **Priority 1**
6. Students should be permitted to ask for the solution of unsigned decimal number's one's complement by the framework. **Priority 1**
7. Students should be permitted to ask for the solution of signed decimal number's one's complement by the framework. **Priority 1**
8. Students should be permitted to ask for the solution of unsigned binary number's one's complement by the framework. **Priority 1**
9. Students should be permitted to ask for the solution of signed binary number's one's complement by the framework. **Priority 1**
10. Students should be permitted to ask for the solution of unsigned decimal number's two's complement by the framework. **Priority 1**
11. Students should be permitted to ask for the solution of signed decimal number's two's complement by the framework. **Priority 1**
12. Students should be permitted to ask for the solution of unsigned binary number's two's complement by the framework. **Priority 1**

Downloaded by HASSAN ALI (hxn112233@gmail.com)

13. Students should be permitted to ask for the solution of signed binary number's two's complement by the framework. **Priority 1**
14. Students should be permitted to ask for queries related to related to binary to decimal conversion by the framework. **Priority 1**
15. Students should be permitted to ask for queries related to decimal to binary conversion by the framework. **Priority 1**
16. Students should be permitted to ask for queries related to number of bits required for an unsigned decimal for its binary representation by the framework. **Priority 1**
17. Students should be permitted to ask for queries related to number of bits required for a signed decimal for its binary representation by the framework. **Priority 1**
18. Students should be permitted to ask for the solution of binary operations in their desired entered number of bits if valid by the framework. **Priority 1**

Information Management

Instructor should be permitted to make changes in NLP. **Priority 1**

Non-Functional Requirements

Accuracy

1. More than 80% of the framework responses should be accurate. **Priority 1**

Fast Response

The server should respond timely over an asked question. **Priority 2**

Portability

1. The bot server should be OS friendly and should support multiple operating systems. **Priority 2**

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

2. The bot server should be hardware friendly and should support multiple hardware. **Priority 2**

Maintainability

The server should be easily maintainable for the developer. **Priority 1**

Ethics

The framework should follow code of ethics and should be GDPR compliant i.e. any kind of information about the user using the chatbot should not be stored or processed by any means. **Priority 1**

Answering Generic Questions/Bot Conversion

This module is based on Amazon Lex. It can answer university related questions as well as generic questions. Amazon Lex receives the user's question via API call from the Flask server. It then calls the specific intent of Lex on basis of keywords by parsing the user's question. An intent represents an action that the user wants to perform. It then either returns the response to server, or call the Amazon Lambda function for further processing. The lambda function calls an external API of Wolfram Alpha for providing answers of generic questions. It confers answers relevant to general discussion, just like any other search engine. The JSON response from api is then converted in to readable format for flask server. Lex also answers the questions related to the university. For that we need to create additional intents for the types of questions user may ask, specific to this university.

Binary Module

While the generic module will concentrate on basic queries of the user. The binary module will specifically answer questions related to Mathematics involving Boolean Algebra, Decimal and Binary number systems. This

Downloaded by HASSAN ALI (hxn112233@gmail.com)

binary module is capable of answering questions on simple Boolean algebra in both decimal and binary formats.

However, before performing any of the functions, the query string is first preprocessed using three operations. At the outset, the string is tokenized using NLTK's built-in word_tokenize()[4] method. Tokenization separates individual words and dumps them into a list. Punctuation, prepositions and other such grammatical features get ignored during this process. Once the list of tokenize words gets filled, it is further filtered down by comparing with another fixed list of keywords. These keywords are synonyms of each other and relevant to the context of Boolean calculations.

The following are the few keywords that are used to distinguish the query's context:

Write, One/One's, Two/Two's, Binary, Decimal, Convert, Difference, Bits, Complement, Represent, Sum etc

After narrowing down the query's context, arguments get parsed next. The binary module at this point can only parse up to 2 arguments. In order to distinguish whether arguments are of binary or decimal nature, check_binary_format() method is called which checks if only 1s and 0s are present in the argument. For example, if an argument is 101101 then check_binary_format()[5] will return true. Contrary If the argument is let's say 10012 then it will return false.

Depending on arguments' format and keywords present in the string, the applicable Boolean operations gets called. On the occasion that the string contains:

---

[4] Check Appendix C

[5] Check Appendix C

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

- “One/ones/one’s” and “complement” then one_complement() method is called which returns 1’s complement.
- “Two/Twos/Two’s” and “complement” then two_complement() method is called which returns 2’s complement of the given argument.
- “Bits” then the binary representation of the string is returned via bit_representation() function.
- “Sum” or “+” symbol, then binary_addition() function is applied.
- “Difference” or “-” symbol, then binary_subtraction() function is applied.
- “Convert” and “to decimal”/ “binary to decimal” then binary_to_decimal() function is called.
- “Convert” and “to binary”/ “decimal to binary” then decimal_to_binary() function is called.

Data Storage[6]

All the questions and their answers given by the chatbot is stored in a database on Atlas Instance. It can be used in future to put into effect machine learning and AI approach, for the bot to render highly meticulous answers for students.

---

[6] Check Appendix F

Downloaded by HASSAN ALI (hxn112233@gmail.com)

CHAPTER 4

FINDINGS AND RESULTS

Overview

In this chapter, the test cases have been applied to each module of the chatbot singly. Alongside this the framework responses have been collected to check whether the preliminary established functional and non-functional requirements were met or not.

Binary Module

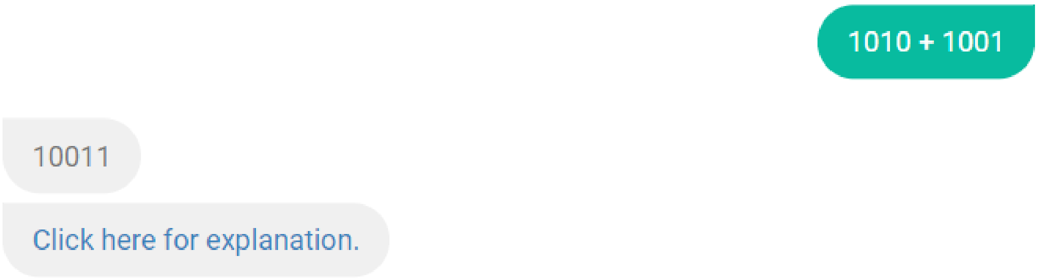
Test Case 01: Chatbot's response for Addition of Binary numbers

Test Example String 1: 1010

Test Example String 2: 1001

Question: What is the result of 1010 + 1001?

Response:



1010 + 1001

10011

Click here for explanation.

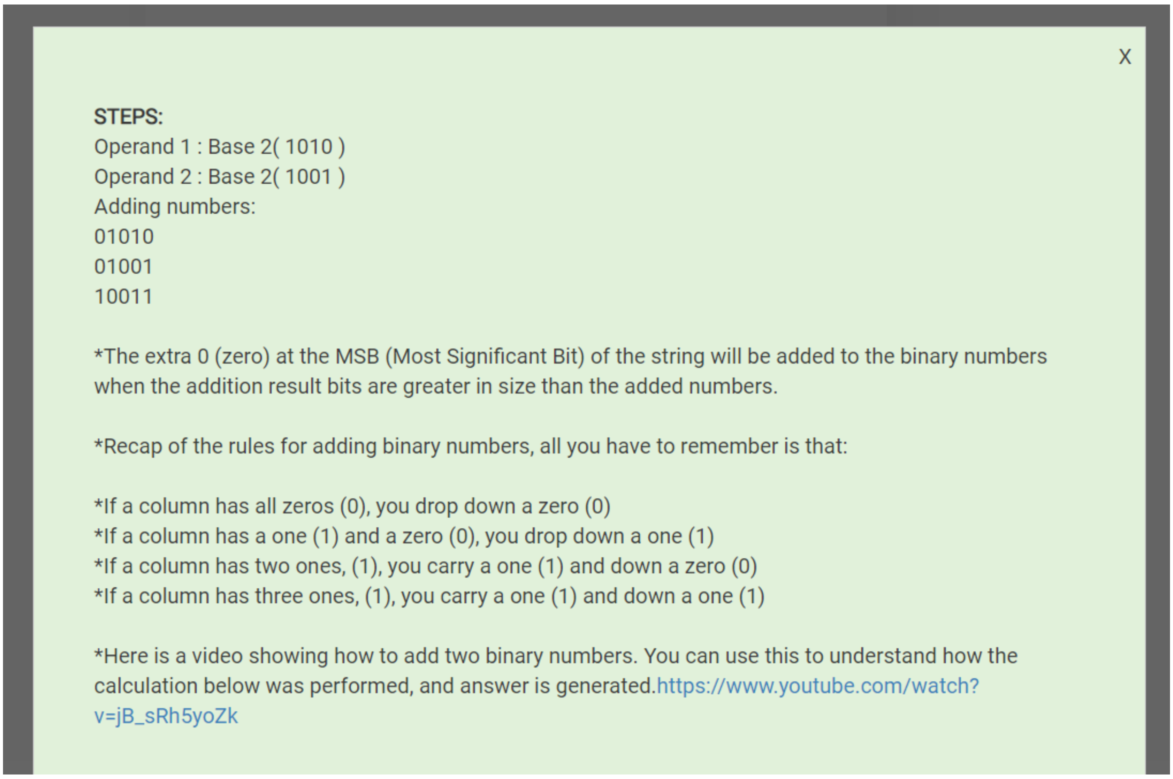
Figure 4.1: Test Case 01: Chatbot's response for Binary Addition

Chatbot's response for binary addition is displayed on the main screen as shown above. Withal, by clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Operand 1 : Base 2(1010)
Operand 2 : Base 2(1001)
Adding numbers:
01010
01001
10011

*The extra 0 (zero) at the MSB (Most Significant Bit) of the string will be added to the binary numbers when the addition result bits are greater in size than the added numbers.

*Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.2: Test Case 01: Explanation box for Binary Addition

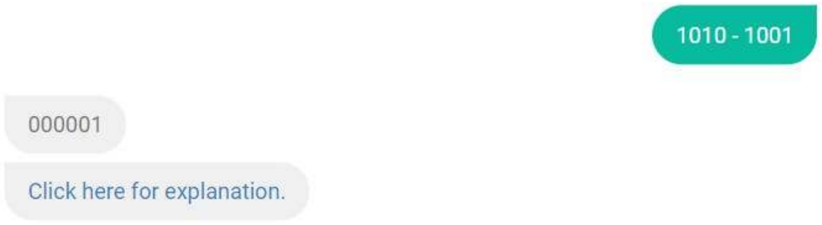
Test Case 02: Chatbot's response for Subtraction of Binary numbers

Test Example String 1: 1010

Test Example String 2: 1001

Question: What is the result of 1010 – 1001?

Response:



1010 - 1001

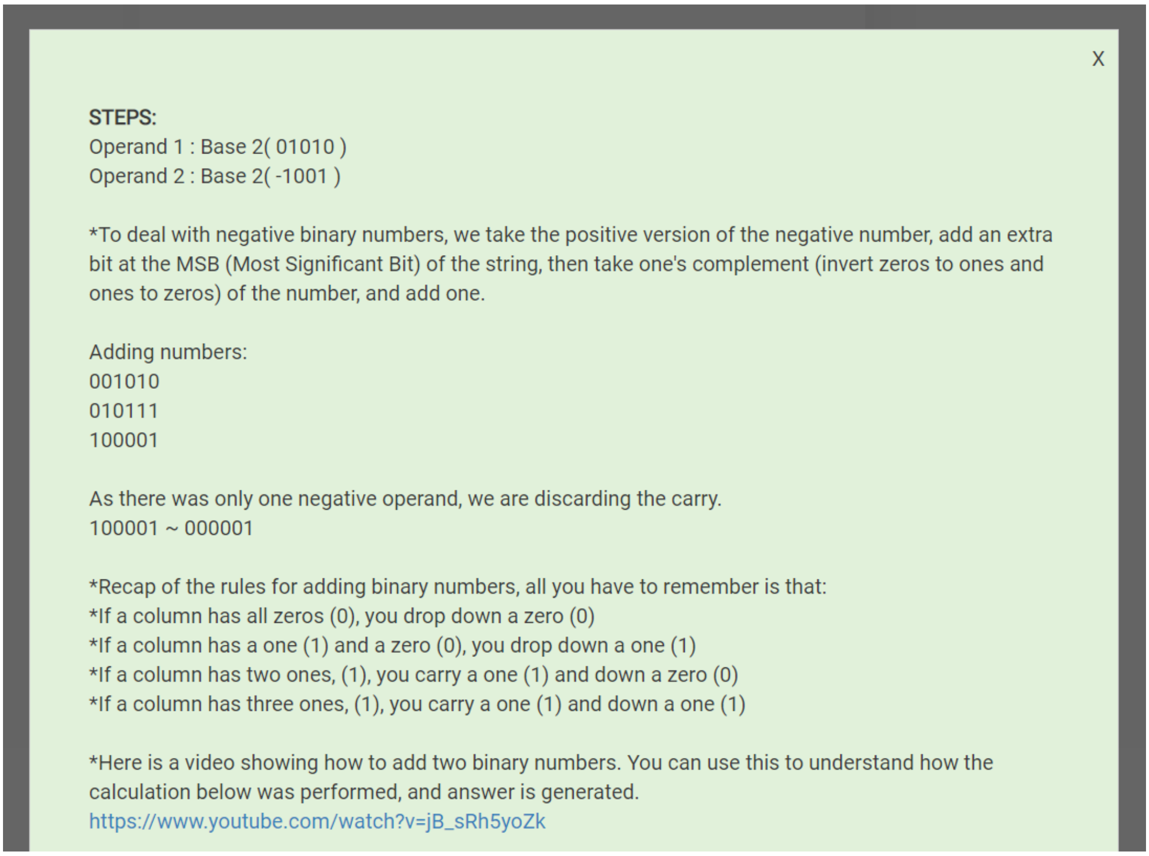
000001

Click here for explanation.

Figure 4.3: Test Case 02: Chatbot's response for Binary Subtraction

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Chatbot’s response for binary subtraction is displayed on the main screen as shown above. Moreover, by clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Operand 1 : Base 2(01010)
Operand 2 : Base 2(-1001)

*To deal with negative binary numbers, we take the positive version of the negative number, add an extra bit at the MSB (Most Significant Bit) of the string, then take one's complement (invert zeros to ones and ones to zeros) of the number, and add one.

Adding numbers:
001010
010111
100001

As there was only one negative operand, we are discarding the carry.
100001 ~ 000001

*Recap of the rules for adding binary numbers, all you have to remember is that:
*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.4: Test Case 02: Explanation box for Binary Subtraction

Test Case 03: Chatbot’s response for Addition of Decimal numbers

Test Example String 1: 69

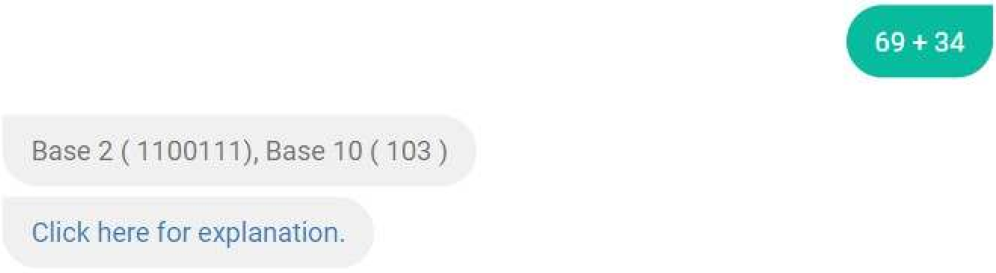
Test Example String 2: 34

Question: What is the result of 69 + 34?

Response:

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



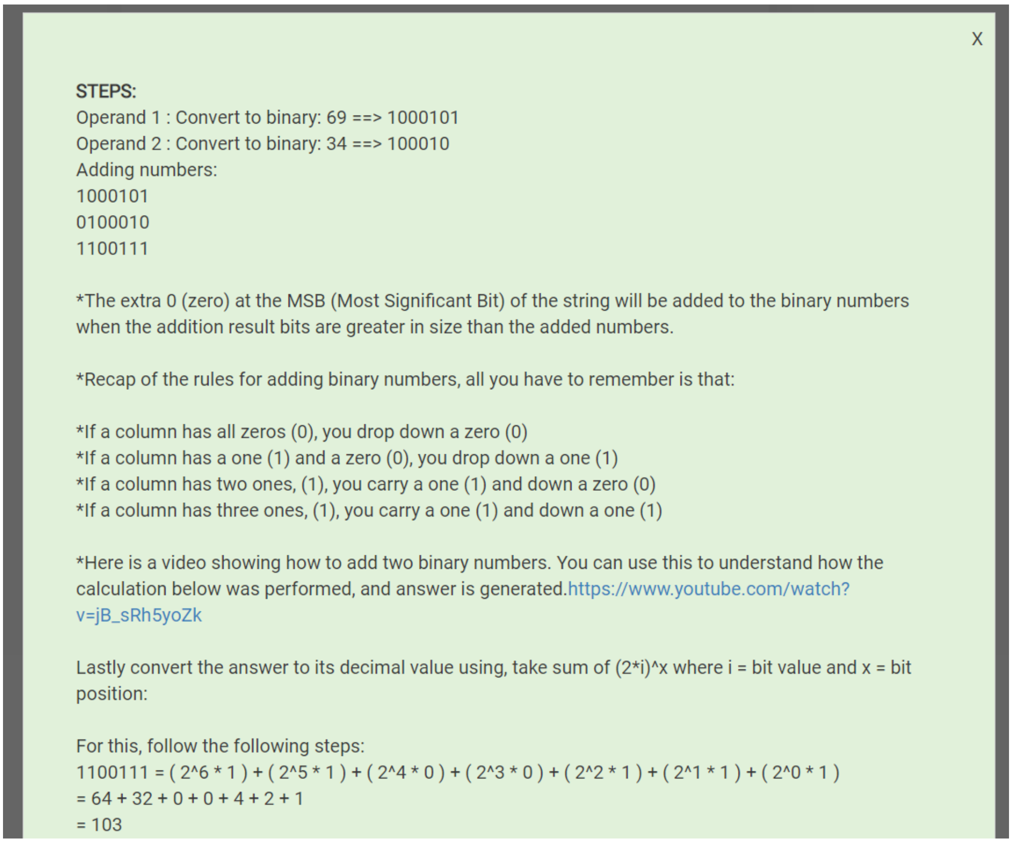
69 + 34

Base 2 (1100111), Base 10 (103)

Click here for explanation.

Figure 4.5: Test Case 03: Chatbot’s response for Decimal Addition

Chatbot’s response for decimal addition is displayed on the main screen as shown above. Additionally, by clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Operand 1 : Convert to binary: 69 ==> 1000101
Operand 2 : Convert to binary: 34 ==> 100010
Adding numbers:
1000101
0100010
1100111

*The extra 0 (zero) at the MSB (Most Significant Bit) of the string will be added to the binary numbers when the addition result bits are greater in size than the added numbers.

*Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.https://www.youtube.com/watch?v=jB_sRh5yoZk

Lastly convert the answer to its decimal value using, take sum of (2*i)^x where i = bit value and x = bit position:

For this, follow the following steps:
1100111 = (2^6 * 1) + (2^5 * 1) + (2^4 * 0) + (2^3 * 0) + (2^2 * 1) + (2^1 * 1) + (2^0 * 1)
= 64 + 32 + 0 + 0 + 4 + 2 + 1
= 103

Figure 4.6: Test Case 03: Explanation box for Decimal Addition

Test Case 04: Chatbot’s response for Subtraction of Decimal numbers

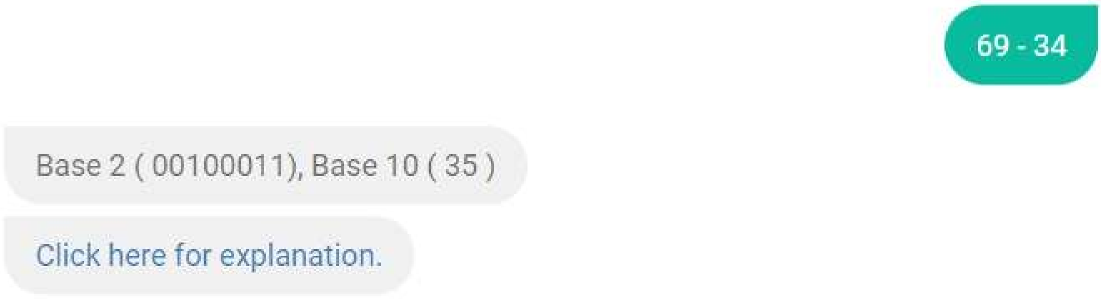
Test Example String 1: 69

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Test Example String 2: 34

Question: What is the result of 69 – 34?

Response:



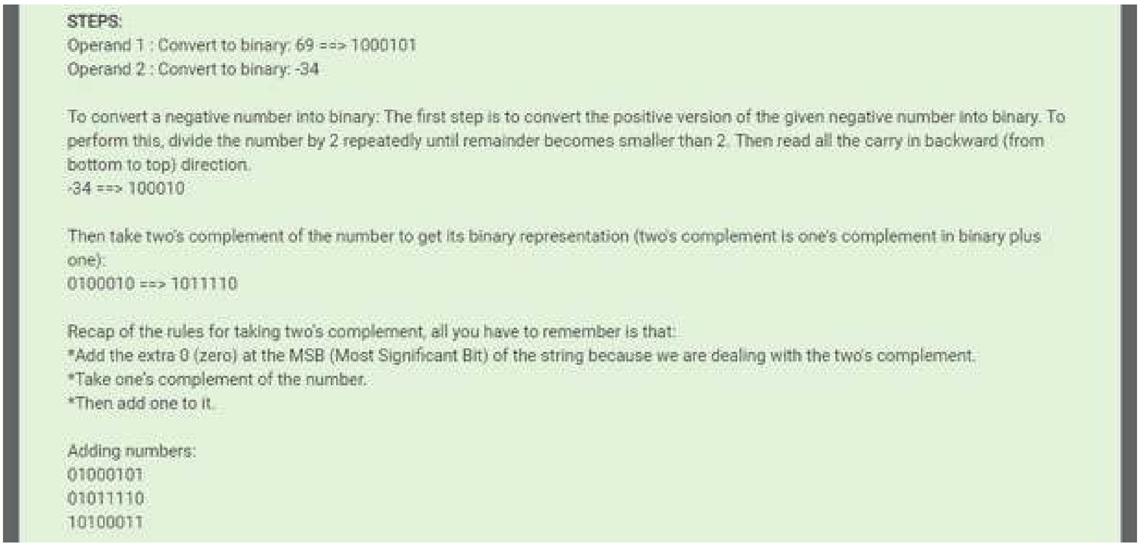
69 - 34

Base 2 (00100011), Base 10 (35)

Click here for explanation.

Figure 4.7: Test Case 04: Chatbot's response for Decimal Subtraction

Chatbot's response for decimal subtraction is displayed on the main screen as shown above. Also, by clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Operand 1 : Convert to binary: 69 ==> 1000101
Operand 2 : Convert to binary: -34

To convert a negative number into binary: The first step is to convert the positive version of the given negative number into binary. To perform this, divide the number by 2 repeatedly until remainder becomes smaller than 2. Then read all the carry in backward (from bottom to top) direction.
-34 ==> 100010

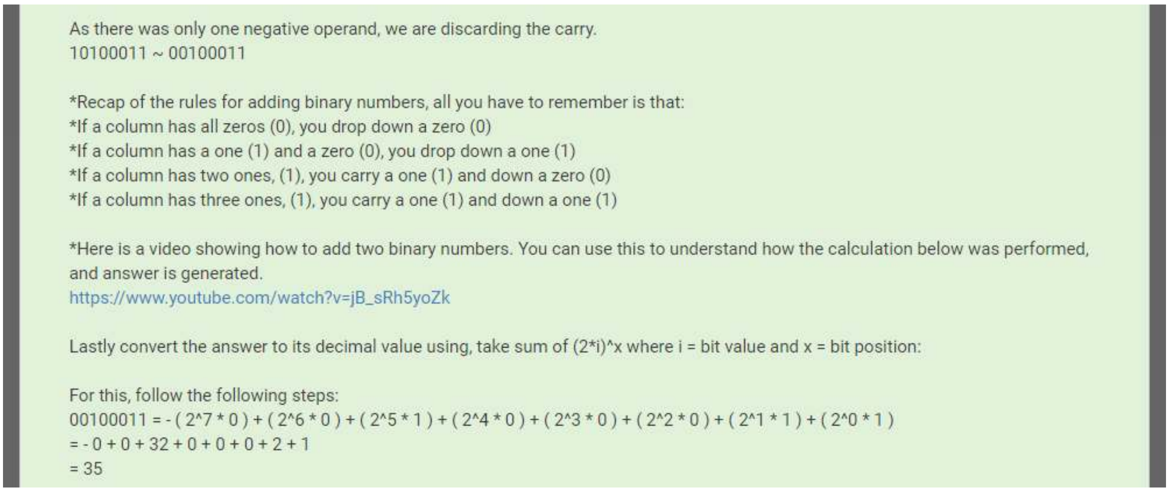
Then take two's complement of the number to get its binary representation (two's complement is one's complement in binary plus one):
0100010 ==> 1011110

Recap of the rules for taking two's complement, all you have to remember is that:
*Add the extra 0 (zero) at the MSB (Most Significant Bit) of the string because we are dealing with the two's complement.
*Take one's complement of the number.
*Then add one to it.

Adding numbers:
01000101
01011110
10100011

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



As there was only one negative operand, we are discarding the carry.
10100011 ~ 00100011

*Recap of the rules for adding binary numbers, all you have to remember is that:
*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Lastly convert the answer to its decimal value using, take sum of (2*i)^x where i = bit value and x = bit position:

For this, follow the following steps:
00100011 = - (2^7 * 0) + (2^6 * 0) + (2^5 * 1) + (2^4 * 0) + (2^3 * 0) + (2^2 * 0) + (2^1 * 1) + (2^0 * 1)
= - 0 + 0 + 32 + 0 + 0 + 0 + 2 + 1
= 35

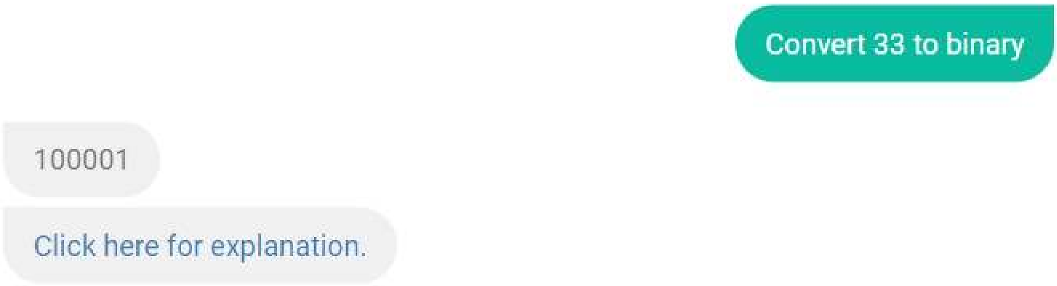
Figure 4.8: Test Case 04: Explanation box for Decimal Subtraction

Test Case 05: Chatbot’s response for decimal to binary conversion

Test Example String: 33

Question: Convert 33 to binary

Response:



Convert 33 to binary

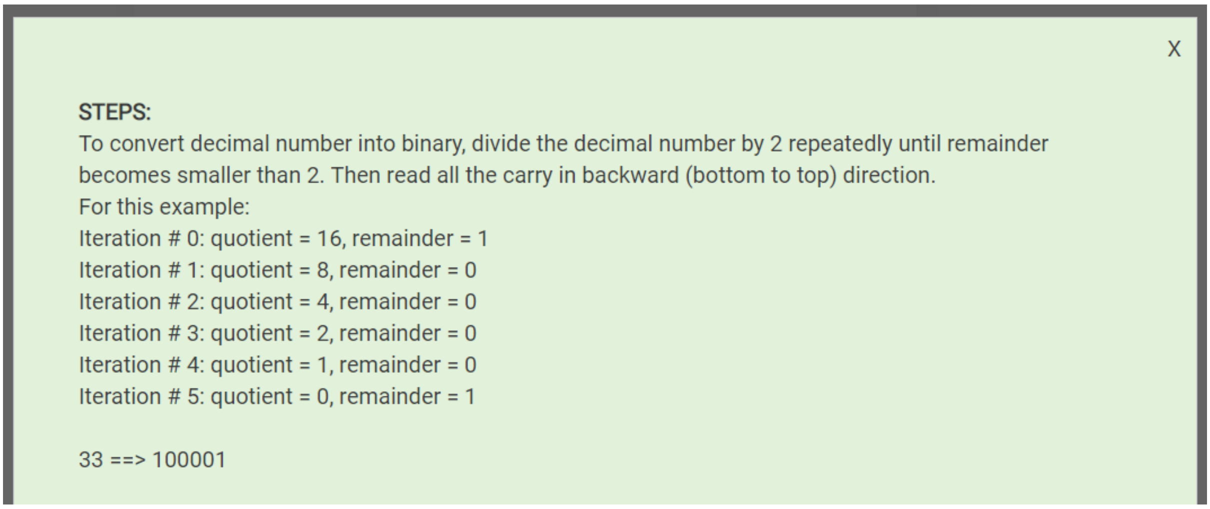
100001

Click here for explanation.

Figure 4.9: Test Case 05: Chatbot’s response for decimal to binary conversion

Chatbot’s response for decimal to binary conversion is displayed on the main screen as shown above. equally, by clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question as shown below.

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
To convert decimal number into binary, divide the decimal number by 2 repeatedly until remainder becomes smaller than 2. Then read all the carry in backward (bottom to top) direction.
For this example:
Iteration # 0: quotient = 16, remainder = 1
Iteration # 1: quotient = 8, remainder = 0
Iteration # 2: quotient = 4, remainder = 0
Iteration # 3: quotient = 2, remainder = 0
Iteration # 4: quotient = 1, remainder = 0
Iteration # 5: quotient = 0, remainder = 1

33 ==> 100001

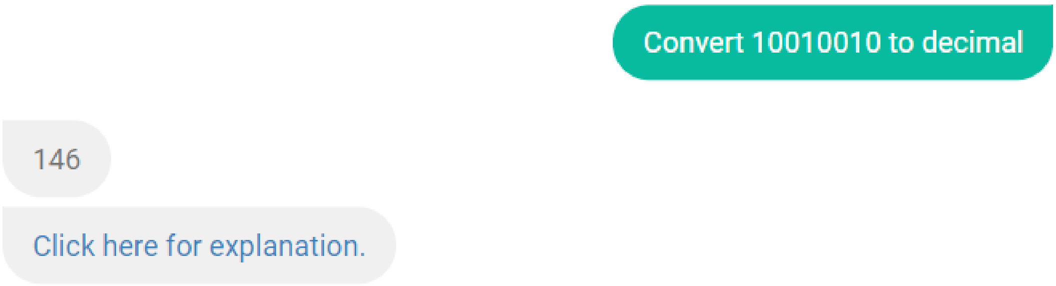
Figure 4.10: Test Case 05: Explanation box for Decimal to binary conversion

Test Case 06: Chatbot's response for binary to decimal conversion

Test Example String: 10010010

Question: Convert 10010010 to decimal

Response:



Convert 10010010 to decimal

146

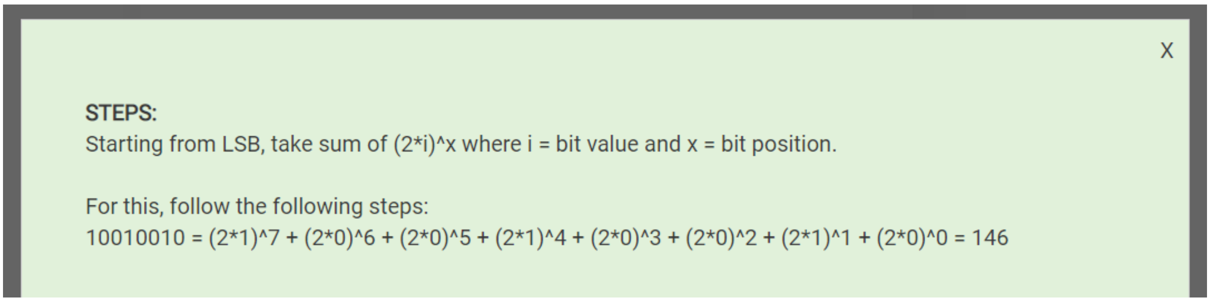
Click here for explanation.

Figure 4.11: Test Case 06: Chatbot's response for binary number to decimal number conversion

Chatbot's response for binary number to decimal number conversion is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question as shown below.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
Starting from LSB, take sum of (2*i)^x where i = bit value and x = bit position.

For this, follow the following steps:
10010010 = (2*1)^7 + (2*0)^6 + (2*0)^5 + (2*1)^4 + (2*0)^3 + (2*0)^2 + (2*1)^1 + (2*0)^0 = 146

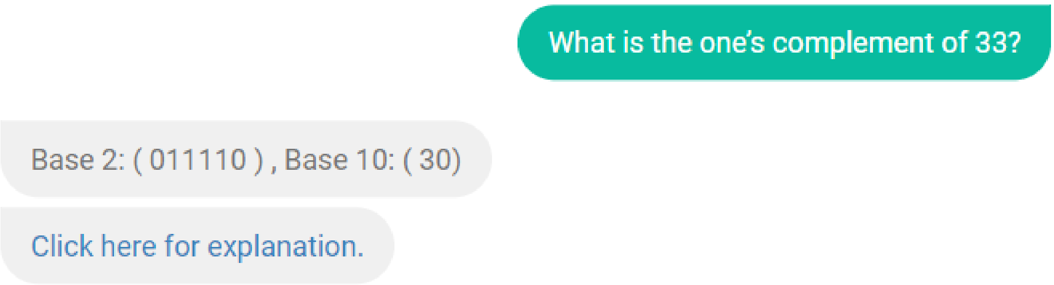
Figure 4.12: Test Case 06: Explanation box for Binary number to decimal number conversion

Test Case 07: Chatbot's response for unsigned decimal number's one's complement

Test Example String: 33

Question: What is the one's complement of 33?

Response:



What is the one's complement of 33?

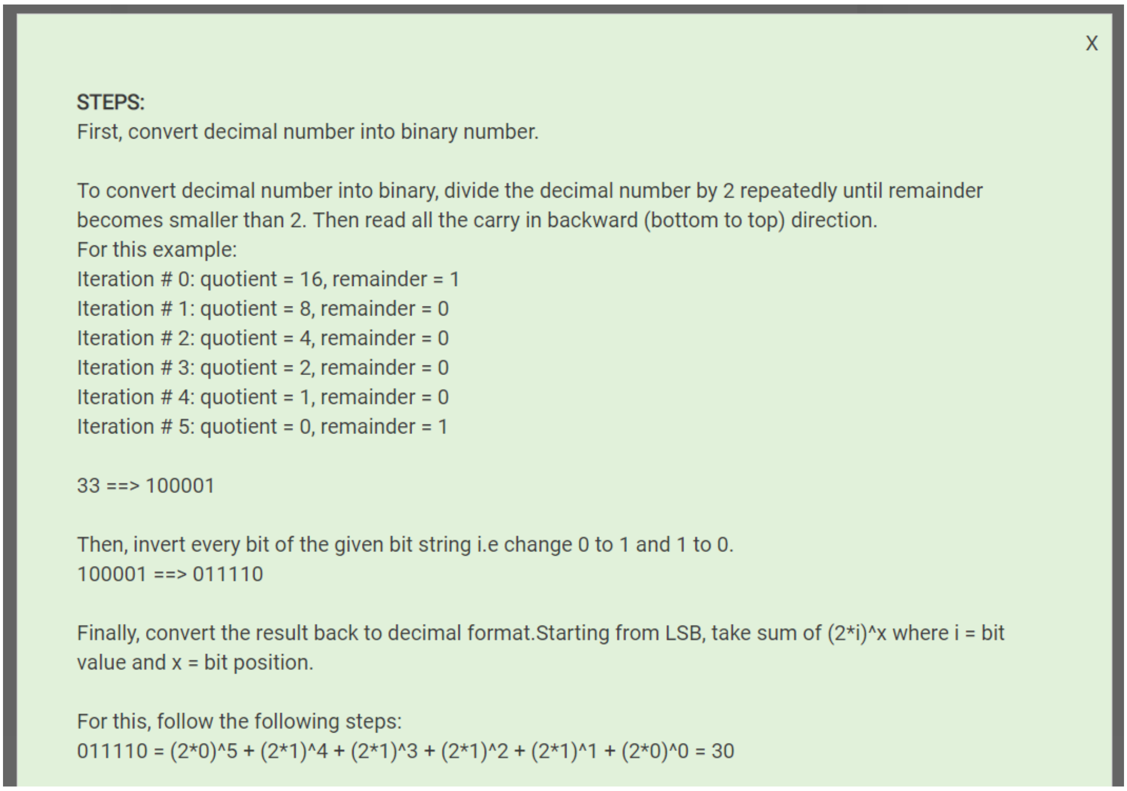
Base 2: (011110) , Base 10: (30)

Click here for explanation.

Figure 4.13: Test Case 07: Chatbot's response for unsigned decimal number's one's complement

Chatbot's response for unsigned decimal number's one's complement is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question as shown below.

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
First, convert decimal number into binary number.

To convert decimal number into binary, divide the decimal number by 2 repeatedly until remainder becomes smaller than 2. Then read all the carry in backward (bottom to top) direction.
For this example:
Iteration # 0: quotient = 16, remainder = 1
Iteration # 1: quotient = 8, remainder = 0
Iteration # 2: quotient = 4, remainder = 0
Iteration # 3: quotient = 2, remainder = 0
Iteration # 4: quotient = 1, remainder = 0
Iteration # 5: quotient = 0, remainder = 1

33 ==> 100001

Then, invert every bit of the given bit string i.e change 0 to 1 and 1 to 0.
100001 ==> 011110

Finally, convert the result back to decimal format.Starting from LSB, take sum of (2*i)^x where i = bit value and x = bit position.

For this, follow the following steps:
011110 = (2*0)^5 + (2*1)^4 + (2*1)^3 + (2*1)^2 + (2*1)^1 + (2*0)^0 = 30

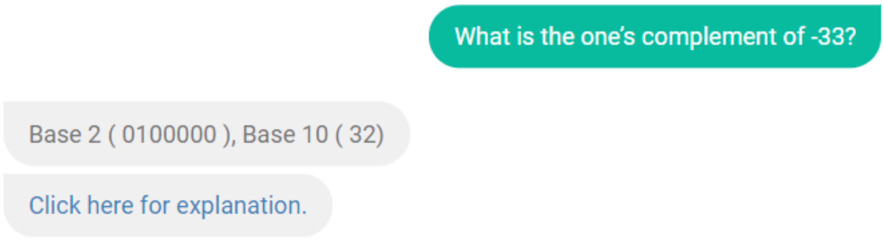
Figure 4.14: Test Case 07: Explanation box for unsigned decimal number's one's complement

Test Case 08: Chatbot's response for signed decimal number's one's complement

Test Example String: -33

Question: What is the one's complement of -33?

Response:



What is the one's complement of -33?

Base 2 (0100000), Base 10 (32)

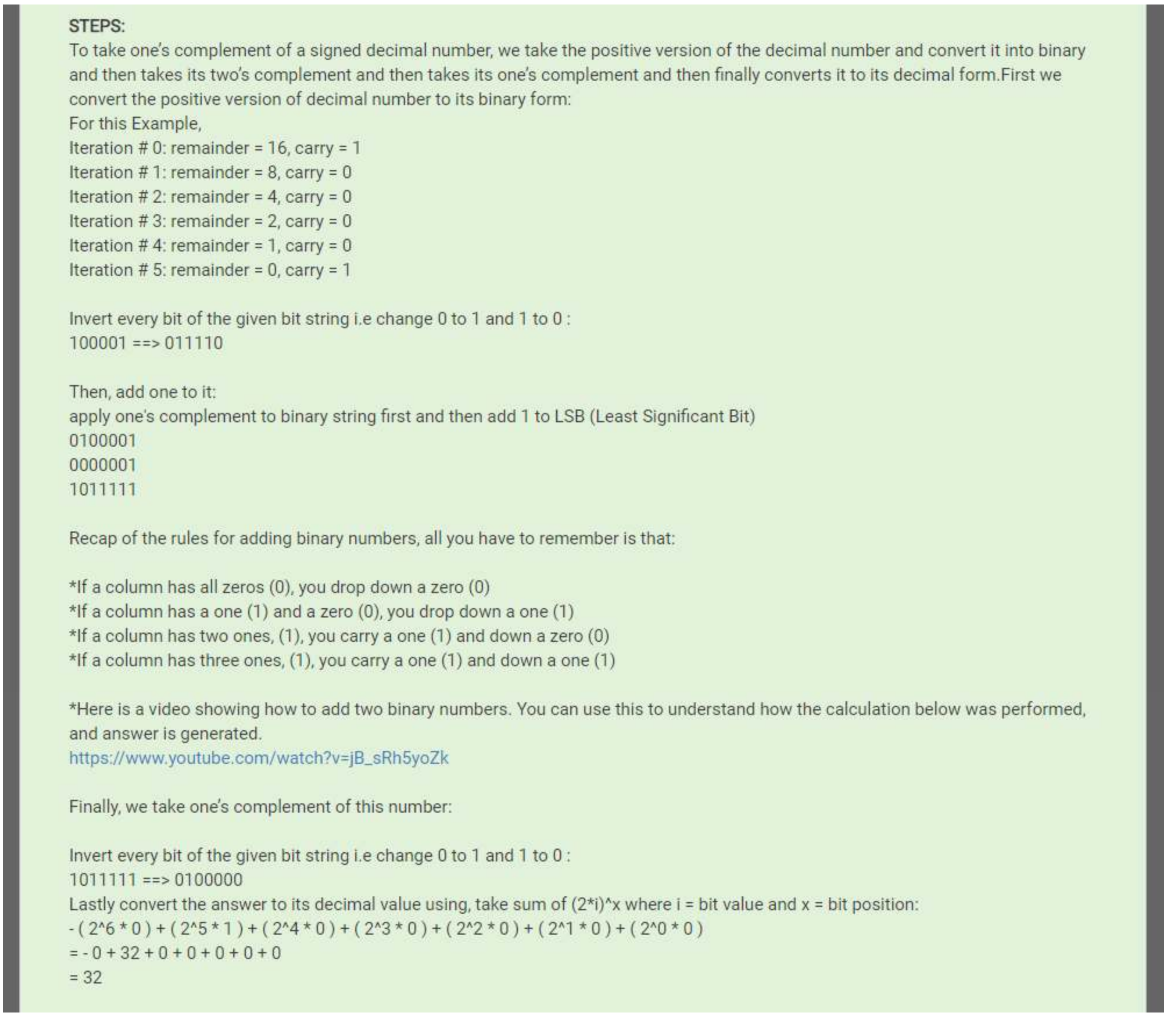
Click here for explanation.

Figure 4.15: Test Case 08: Chatbot's response for signed decimal number's one's complement

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Chatbot’s response for signed decimal number’s one’s complement is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
To take one's complement of a signed decimal number, we take the positive version of the decimal number and convert it into binary and then takes its two's complement and then takes its one's complement and then finally converts it to its decimal form.First we convert the positive version of decimal number to its binary form:
For this Example,
Iteration # 0: remainder = 16, carry = 1
Iteration # 1: remainder = 8, carry = 0
Iteration # 2: remainder = 4, carry = 0
Iteration # 3: remainder = 2, carry = 0
Iteration # 4: remainder = 1, carry = 0
Iteration # 5: remainder = 0, carry = 1

Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :
100001 ==> 011110

Then, add one to it:
apply one's complement to binary string first and then add 1 to LSB (Least Significant Bit)
0100001
0000001
1011111

Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Finally, we take one's complement of this number:

Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :
1011111 ==> 0100000
Lastly convert the answer to its decimal value using, take sum of (2*i)^x where i = bit value and x = bit position:
- (2^6 * 0) + (2^5 * 1) + (2^4 * 0) + (2^3 * 0) + (2^2 * 0) + (2^1 * 0) + (2^0 * 0)
= - 0 + 32 + 0 + 0 + 0 + 0 + 0
= 32

Figure 4.16: Test Case 08: Explanation box for signed decimal number’s one’s complement

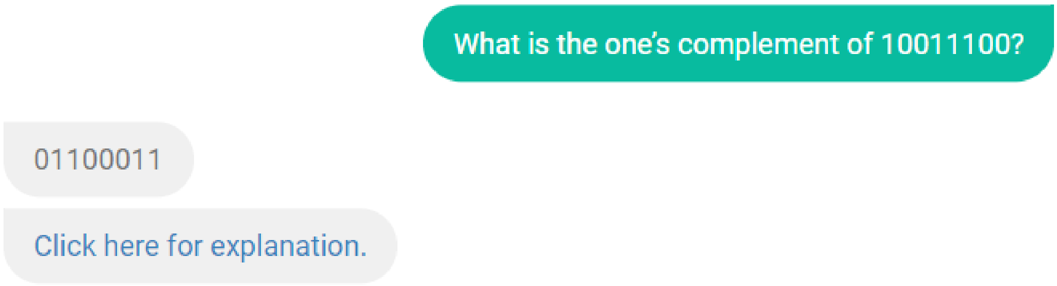
Test Case 09: Chatbot’s response for unsigned binary number’s one’s complement

Test Example String: 10011100

Question: What is the one’s complement of 10011100?

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)

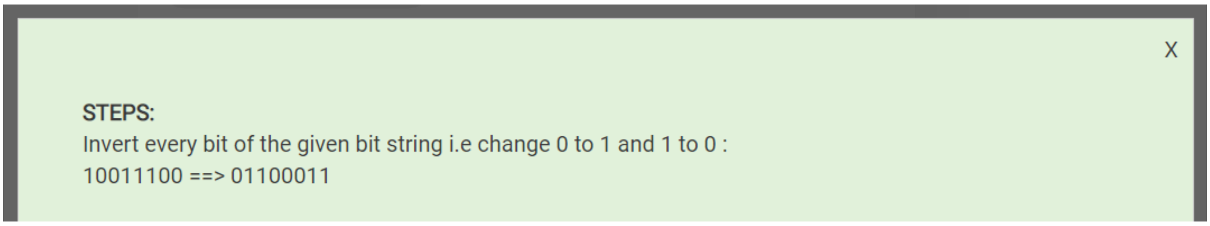

What is the one's complement of 10011100?

01100011

Click here for explanation.

Figure 4.17: Test Case 09: Chatbot's response for unsigned binary number's one's complement

Chatbot's response for unsigned binary number's one's complement is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question as shown below.



STEPS:
Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :
10011100 ==> 01100011

Figure 4.18: Test Case 09: Explanation box for unsigned binary number's one's complement

Test Case 10: Chatbot's response for signed binary number's one's complement

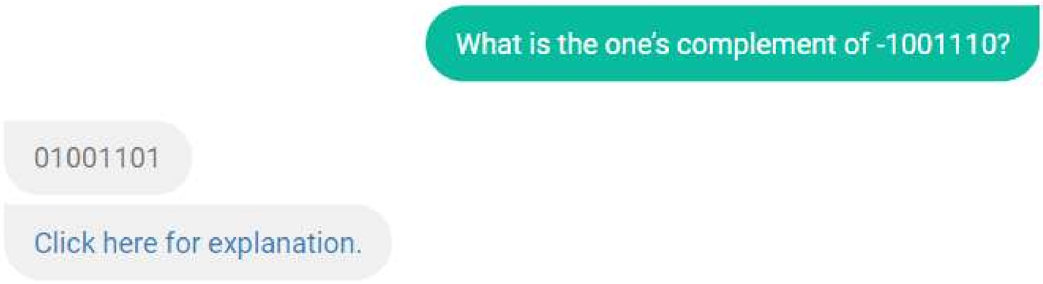
Test Example String: -1001110

Question: What is the one's complement of -1001110?

Response:

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

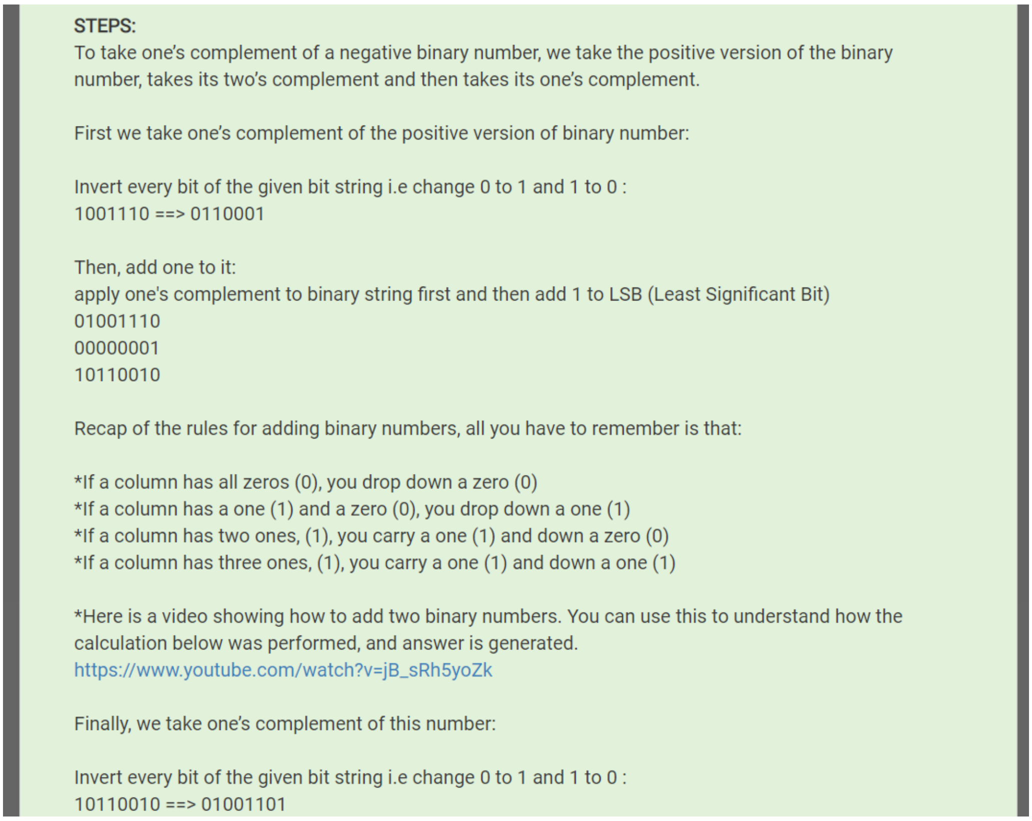

What is the one's complement of -1001110?

01001101

Click here for explanation.

Figure 4.19: Test Case 10: Chatbot's response for signed binary number's one's complement

Chatbot's response for signed binary number's one's complement is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.


STEPS:
To take one's complement of a negative binary number, we take the positive version of the binary number, takes its two's complement and then takes its one's complement.

First we take one's complement of the positive version of binary number:

Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :
1001110 ==> 0110001

Then, add one to it:
apply one's complement to binary string first and then add 1 to LSB (Least Significant Bit)
01001110
00000001
10110010

Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Finally, we take one's complement of this number:

Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :
10110010 ==> 01001101

Figure 4.20: Test Case 10: Explanation box for signed binary number's one's complement

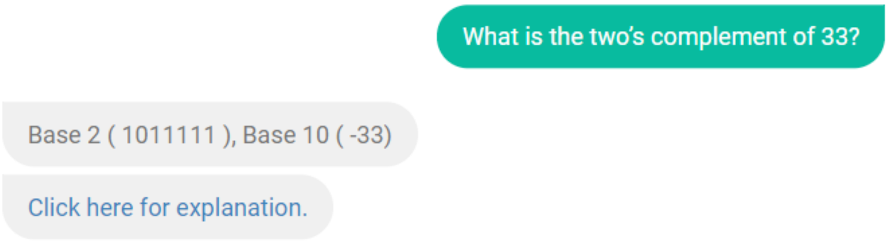
Downloaded by HASSAN ALI (hxn112233@gmail.com)

Test Case 11: Chatbot’s response for unsigned decimal number’s two’s complement

Test Example String: 33

Question: What is the two’s complement of 33?

Response:



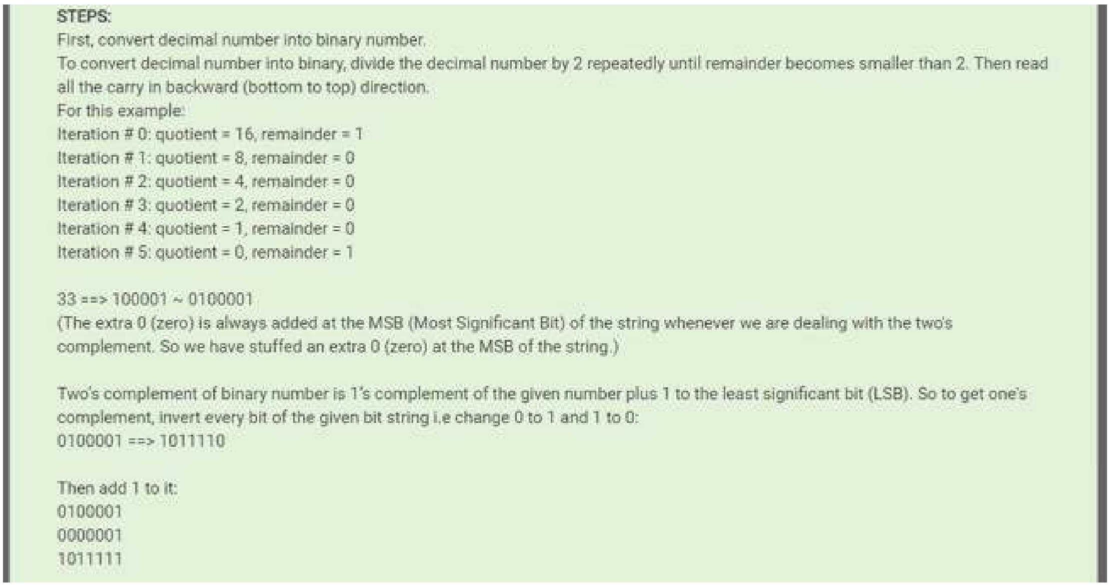
What is the two’s complement of 33?

Base 2 (1011111), Base 10 (-33)

Click here for explanation.

Figure 4.21: Test Case 11: Chatbot’s response for unsigned decimal number’s two’s complement

Chatbot’s response for unsigned decimal number’s two’s complement is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
First, convert decimal number into binary number.
To convert decimal number into binary, divide the decimal number by 2 repeatedly until remainder becomes smaller than 2. Then read all the carry in backward (bottom to top) direction.
For this example:
Iteration # 0: quotient = 16, remainder = 1
Iteration # 1: quotient = 8, remainder = 0
Iteration # 2: quotient = 4, remainder = 0
Iteration # 3: quotient = 2, remainder = 0
Iteration # 4: quotient = 1, remainder = 0
Iteration # 5: quotient = 0, remainder = 1

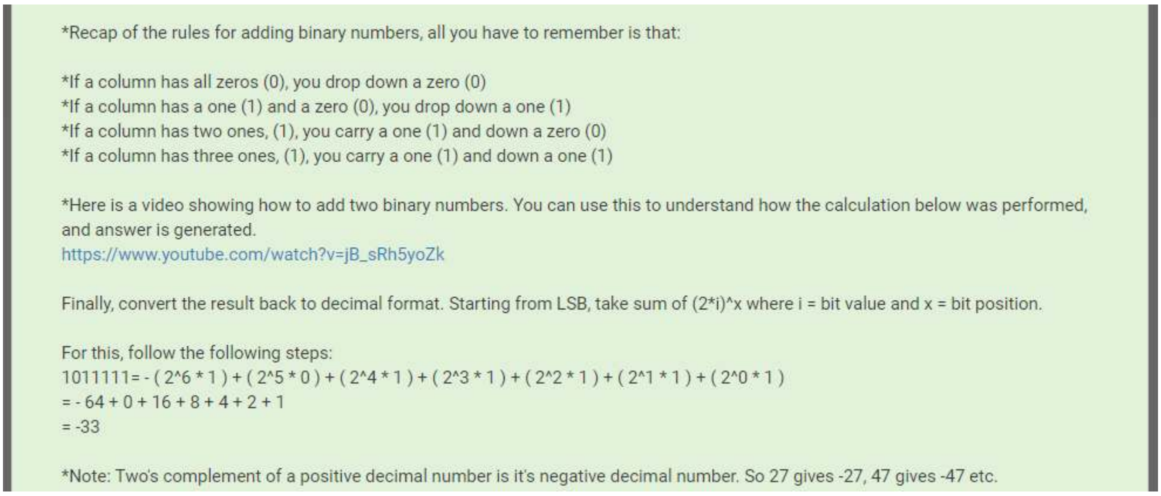
33 ==> 100001 ~ 0100001
(The extra 0 (zero) is always added at the MSB (Most Significant Bit) of the string whenever we are dealing with the two's complement. So we have stuffed an extra 0 (zero) at the MSB of the string.)

Two's complement of binary number is 1's complement of the given number plus 1 to the least significant bit (LSB). So to get one's complement, invert every bit of the given bit string i.e change 0 to 1 and 1 to 0:
0100001 ==> 1011110

Then add 1 to it:
0100001
0000001
1011111

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



*Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Finally, convert the result back to decimal format. Starting from LSB, take sum of (2*i)^x where i = bit value and x = bit position.

For this, follow the following steps:
1011111= - (2^6 * 1) + (2^5 * 0) + (2^4 * 1) + (2^3 * 1) + (2^2 * 1) + (2^1 * 1) + (2^0 * 1)
= - 64 + 0 + 16 + 8 + 4 + 2 + 1
= -33

*Note: Two's complement of a positive decimal number is it's negative decimal number. So 27 gives -27, 47 gives -47 etc.

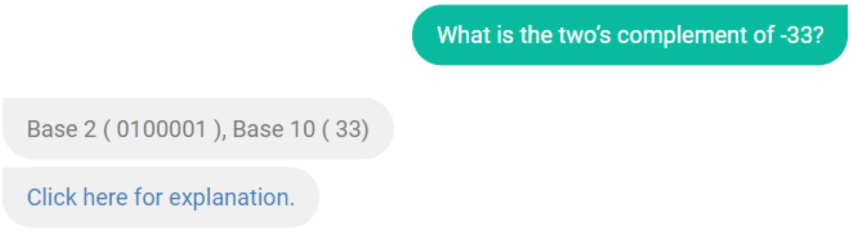
Figure 4.22: Test Case 11: Explanation box for unsigned decimal number's two's complement

Test Case 12: Chatbot's response for signed decimal number's two's complement

Test Example String: -33

Question: What is the two's complement of -33?

Response:



What is the two's complement of -33?

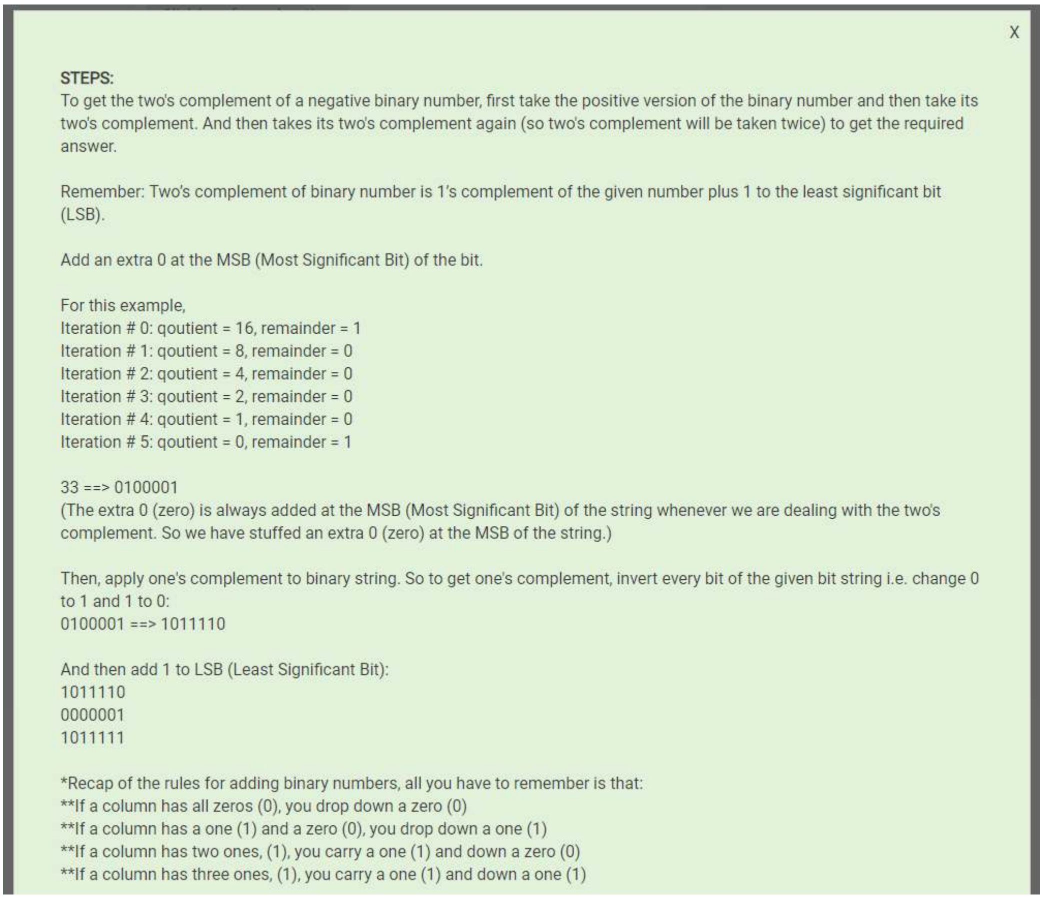
Base 2 (0100001), Base 10 (33)

Click here for explanation.

Figure 4.23: Test Case 12: Chatbot's response for signed decimal number's two's complement

Chatbot's response for signed decimal number's two's complement is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
To get the two's complement of a negative binary number, first take the positive version of the binary number and then take its two's complement. And then takes its two's complement again (so two's complement will be taken twice) to get the required answer.

Remember: Two's complement of binary number is 1's complement of the given number plus 1 to the least significant bit (LSB).

Add an extra 0 at the MSB (Most Significant Bit) of the bit.

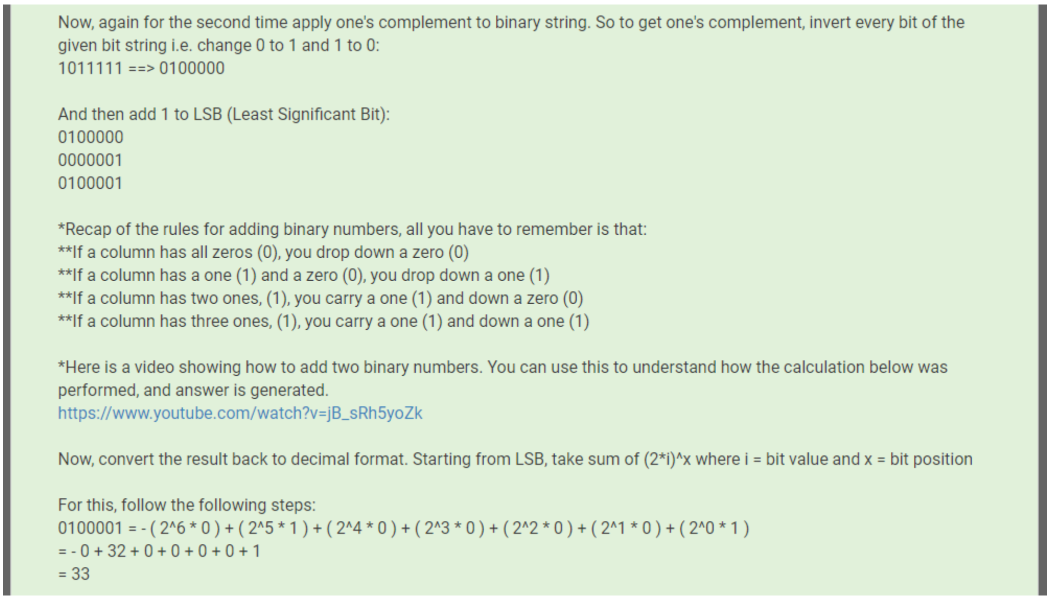
For this example,
Iteration # 0: qoutient = 16, remainder = 1
Iteration # 1: qoutient = 8, remainder = 0
Iteration # 2: qoutient = 4, remainder = 0
Iteration # 3: qoutient = 2, remainder = 0
Iteration # 4: qoutient = 1, remainder = 0
Iteration # 5: qoutient = 0, remainder = 1

33 ==> 0100001
(The extra 0 (zero) is always added at the MSB (Most Significant Bit) of the string whenever we are dealing with the two's complement. So we have stuffed an extra 0 (zero) at the MSB of the string.)

Then, apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
0100001 ==> 1011110

And then add 1 to LSB (Least Significant Bit):
1011110
0000001
1011111

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)



Now, again for the second time apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
1011111 ==> 0100000

And then add 1 to LSB (Least Significant Bit):
0100000
0000001
0100001

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Now, convert the result back to decimal format. Starting from LSB, take sum of (2*i)^x where i = bit value and x = bit position

For this, follow the following steps:
0100001 = - (2^6 * 0) + (2^5 * 1) + (2^4 * 0) + (2^3 * 0) + (2^2 * 0) + (2^1 * 0) + (2^0 * 1)
= - 0 + 32 + 0 + 0 + 0 + 0 + 1
= 33

Figure 4.24: Test Case 12: Explanation box for signed decimal number's two's complement

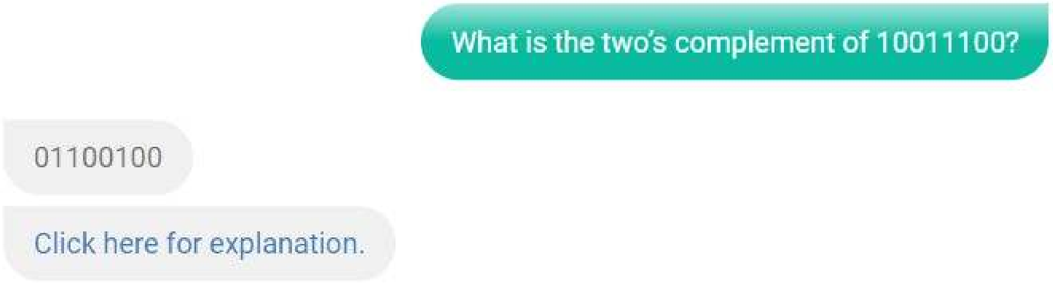
This document is available free of charge on studocu
Downloaded by HASSAN ALI (hxn112233@gmail.com)

Test Case 13: Chatbot’s response for unsigned binary number’s two’s complement

Test Example String: 10011100

Question: What is the two’s complement of 10011100?

Response:



What is the two’s complement of 10011100?

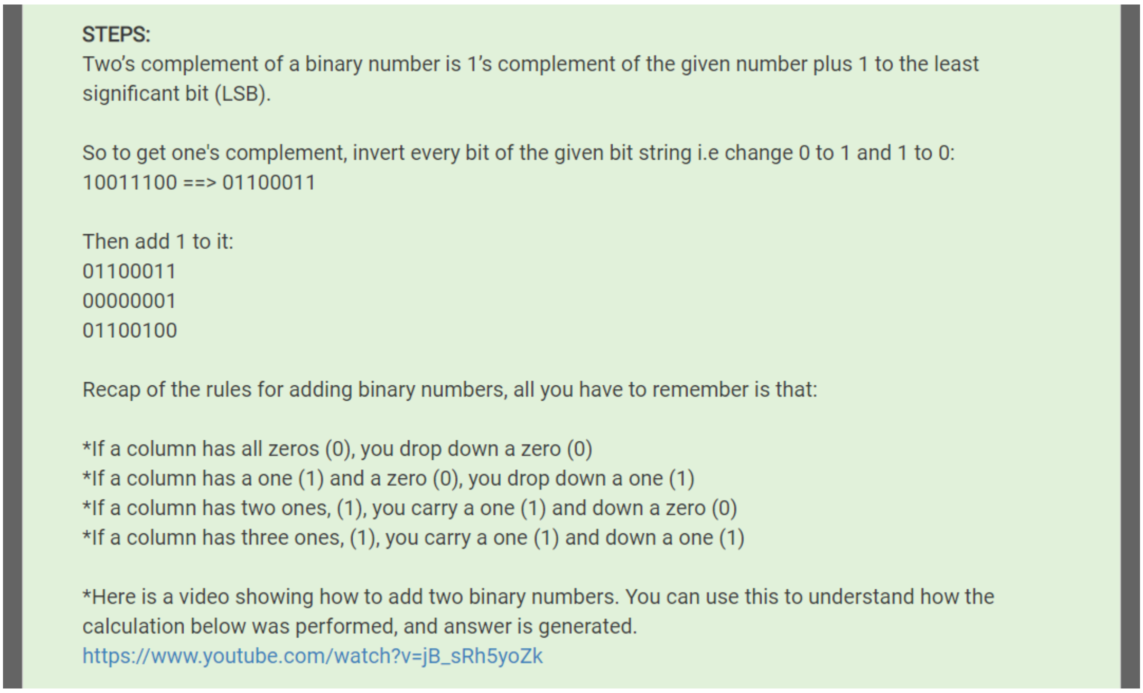
01100100

Click here for explanation.

Figure 4.25: Test Case 13: Chatbot’s response for unsigned binary number’s two’s complement

Chatbot’s response for unsigned binary number’s two’s complement is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
Two's complement of a binary number is 1's complement of the given number plus 1 to the least significant bit (LSB).

So to get one's complement, invert every bit of the given bit string i.e change 0 to 1 and 1 to 0:
10011100 ==> 01100011

Then add 1 to it:
01100011
00000001
01100100

Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

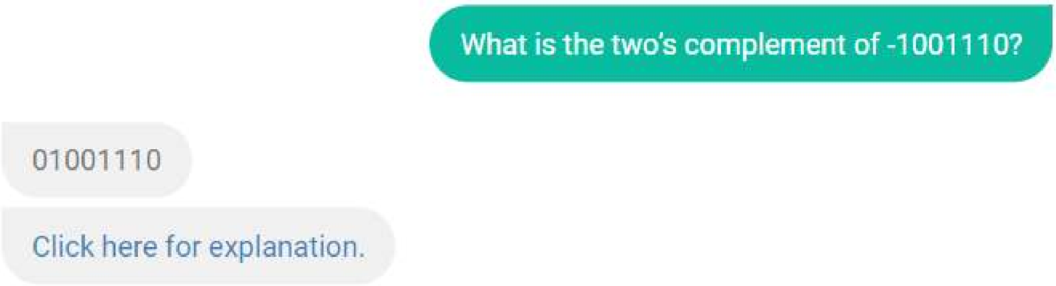
Figure 4.26: Test Case 13: Explanation box for unsigned binary number's two's complement

Test Case 14: Chatbot's response for signed binary number's two's complement

Test Example String: -1001110

Question: What is the two's complement of -1001110?

Response:



What is the two's complement of -1001110?

01001110

Click here for explanation.

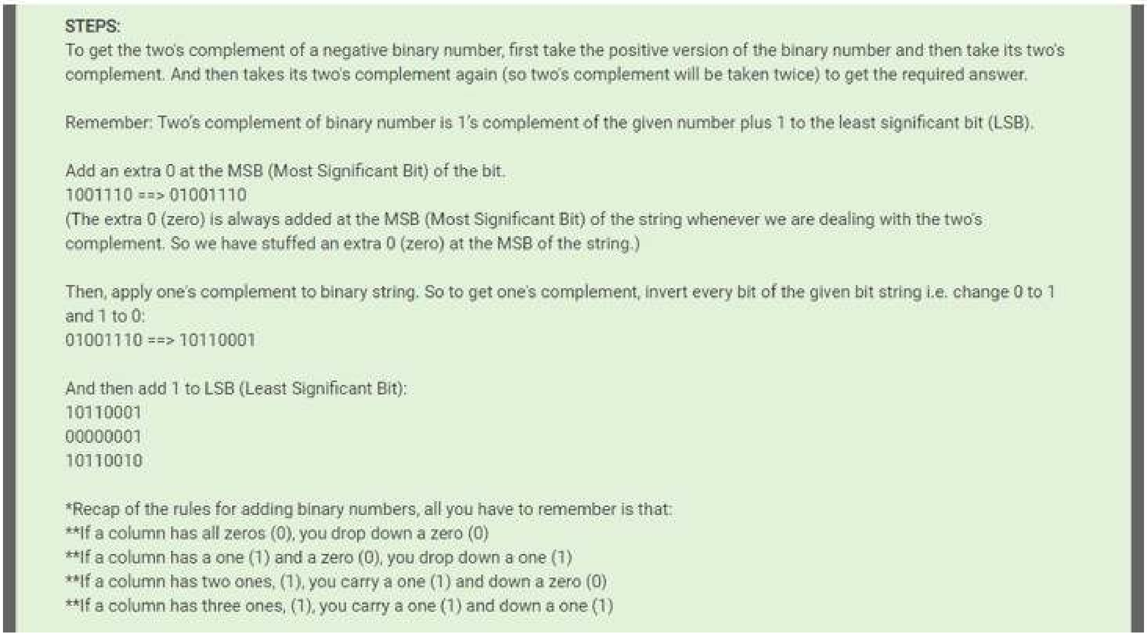
Figure 4.27: Test Case 14: Chatbot's response for signed binary number's two's complement

Chatbot's response for signed binary number's two's complement is displayed on the main screen as shown above. By clicking on the 'Click

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:

To get the two's complement of a negative binary number, first take the positive version of the binary number and then take its two's complement. And then takes its two's complement again (so two's complement will be taken twice) to get the required answer.

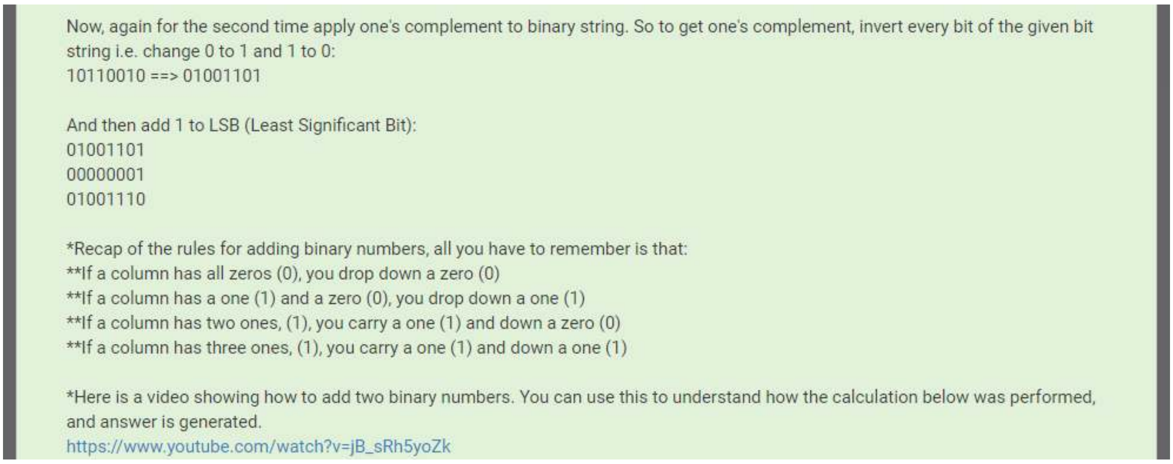
Remember: Two's complement of binary number is 1's complement of the given number plus 1 to the least significant bit (LSB).

Add an extra 0 at the MSB (Most Significant Bit) of the bit.
1001110 ==> 01001110
(The extra 0 (zero) is always added at the MSB (Most Significant Bit) of the string whenever we are dealing with the two's complement. So we have stuffed an extra 0 (zero) at the MSB of the string.)

Then, apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
01001110 ==> 10110001

And then add 1 to LSB (Least Significant Bit):
10110001
00000001
10110010

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)



Now, again for the second time apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
10110010 ==> 01001101

And then add 1 to LSB (Least Significant Bit):
01001101
00000001
01001110

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.28: Test Case 14: Explanation box for signed binary number's two's complement

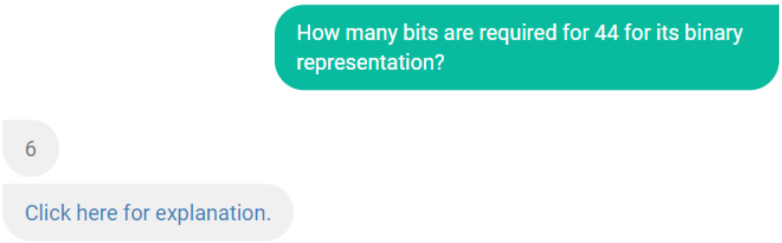
Test Case 15: Chatbot's response for unsigned decimal's binary representation bits requirement

Test Example String: 44

Question: How many bits are required for 44 for its binary representation?

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)

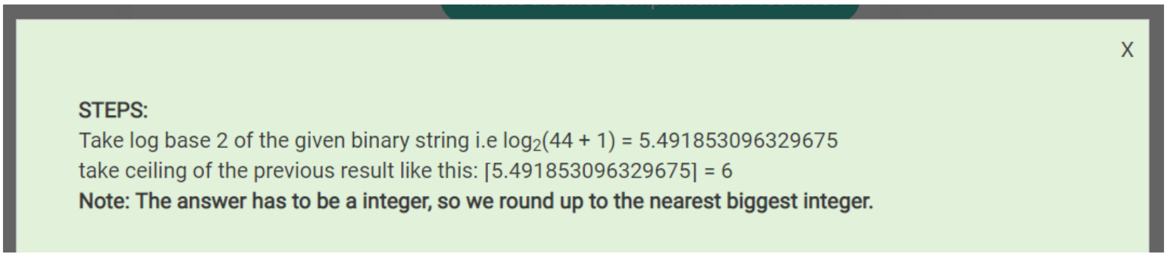


How many bits are required for 44 for its binary representation?

6

Figure 4.29: Test Case 15: Chatbot’s response for unsigned decimal’s binary representation bits requirement

Chatbot’s response for unsigned decimal’s binary representation bits requirement is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question as shown below.



STEPS:
Take log base 2 of the given binary string i.e $\log_2(44 + 1) = 5.491853096329675$
take ceiling of the previous result like this: $\lceil 5.491853096329675 \rceil = 6$
Note: The answer has to be a integer, so we round up to the nearest biggest integer.

Figure 4.30: Test Case 15: Explanation box response for unsigned decimal’s binary representation bits requirement

Test Case 16: Chatbot’s response for signed decimal’s binary representation bits requirement

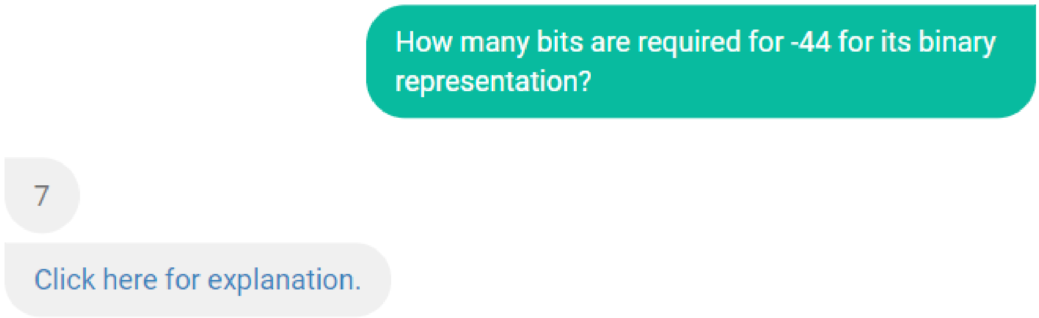
Test Example String: -44

Question: How many bits are required for -44 for its binary representation?

Response:

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)


How many bits are required for -44 for its binary representation?

7

Click here for explanation.

Figure 4.31: Test Case 16: Chatbot’s response for signed decimal’s binary representation bits requirement

Chatbot’s response for signed decimal’s binary representation bits requirement is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question as shown below.

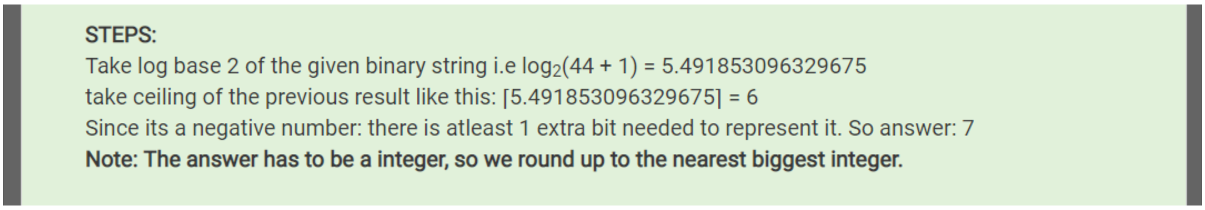

STEPS:
Take log base 2 of the given binary string i.e $\log_2(44 + 1) = 5.491853096329675$
take ceiling of the previous result like this: $\lceil 5.491853096329675 \rceil = 6$
Since its a negative number: there is atleast 1 extra bit needed to represent it. So answer: 7
Note: The answer has to be a integer, so we round up to the nearest biggest integer.

Figure 4.32: Test Case 16: Explanation box for signed decimal’s binary representation bits requirement

Test Case 17: User Input Bits Functionality

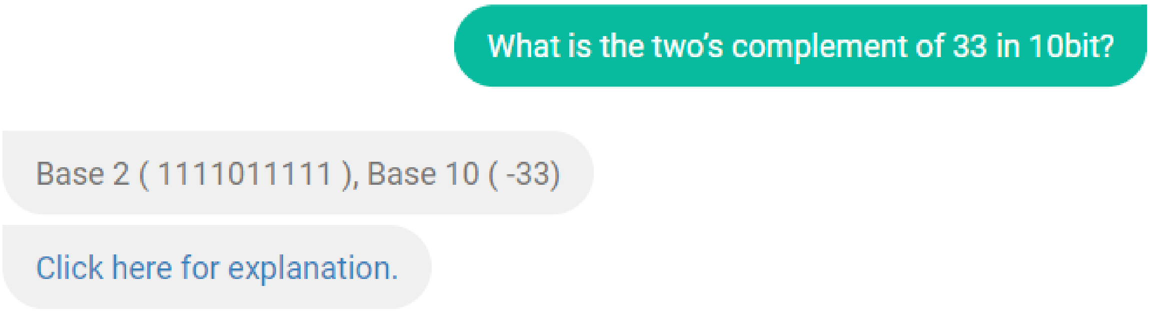
Subcase 1: Chatbot’s response for unsigned decimal number’s two’s complement in user-specified bits

Test Example String: 33

Question: What is the two’s complement of 33 in 10bit?

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)



What is the two's complement of 33 in 10bit?

Base 2 (1111011111), Base 10 (-33)

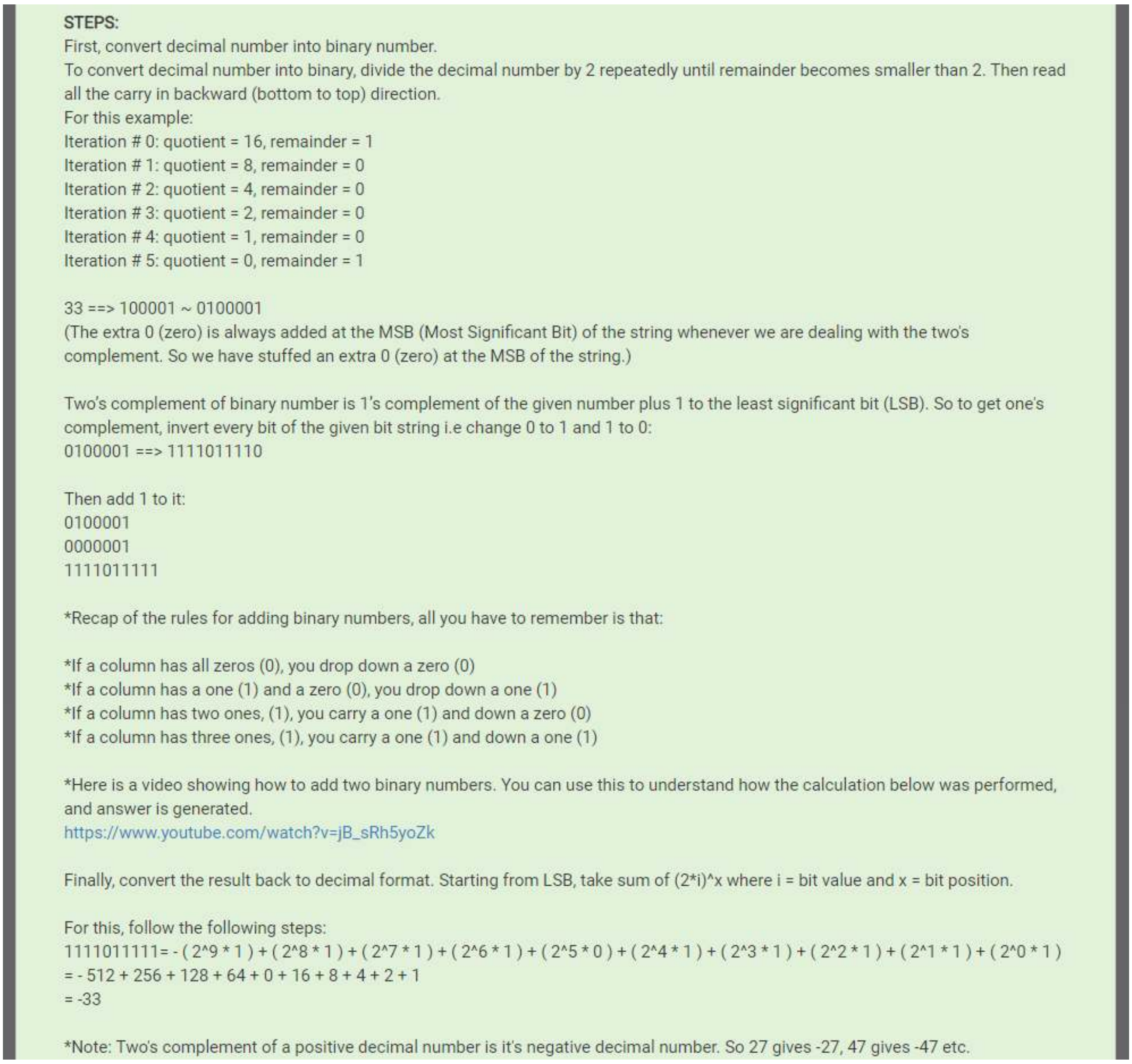
Click here for explanation.

Figure 4.33: Test Case 17 Subcase 1: Chatbot's response for unsigned decimal number's two's complement in user-specified bits

Chatbot's response for unsigned decimal number's two's complement in user-specified bits is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
First, convert decimal number into binary number.
To convert decimal number into binary, divide the decimal number by 2 repeatedly until remainder becomes smaller than 2. Then read all the carry in backward (bottom to top) direction.
For this example:
Iteration # 0: quotient = 16, remainder = 1
Iteration # 1: quotient = 8, remainder = 0
Iteration # 2: quotient = 4, remainder = 0
Iteration # 3: quotient = 2, remainder = 0
Iteration # 4: quotient = 1, remainder = 0
Iteration # 5: quotient = 0, remainder = 1

33 ==> 100001 ~ 0100001
(The extra 0 (zero) is always added at the MSB (Most Significant Bit) of the string whenever we are dealing with the two's complement. So we have stuffed an extra 0 (zero) at the MSB of the string.)

Two's complement of binary number is 1's complement of the given number plus 1 to the least significant bit (LSB). So to get one's complement, invert every bit of the given bit string i.e change 0 to 1 and 1 to 0:
0100001 ==> 1111011110

Then add 1 to it:
0100001
0000001
1111011111

*Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Finally, convert the result back to decimal format. Starting from LSB, take sum of (2*i)^x where i = bit value and x = bit position.

For this, follow the following steps:
1111011111= - (2^9 * 1) + (2^8 * 1) + (2^7 * 1) + (2^6 * 1) + (2^5 * 0) + (2^4 * 1) + (2^3 * 1) + (2^2 * 1) + (2^1 * 1) + (2^0 * 1)
= - 512 + 256 + 128 + 64 + 0 + 16 + 8 + 4 + 2 + 1
= -33

*Note: Two's complement of a positive decimal number is it's negative decimal number. So 27 gives -27, 47 gives -47 etc.

Figure 4.34: Test Case 17 Subcase 1: Explanation box for unsigned decimal number's two's complement in user-specified bits

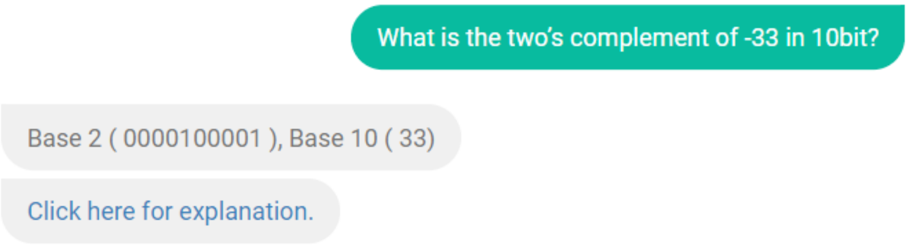
Subcase 2: Chatbot's response for signed decimal number's two's complement in user-specified bits

Test Example String: -33

Question: What is the two's complement of -33 in 10bit?

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)

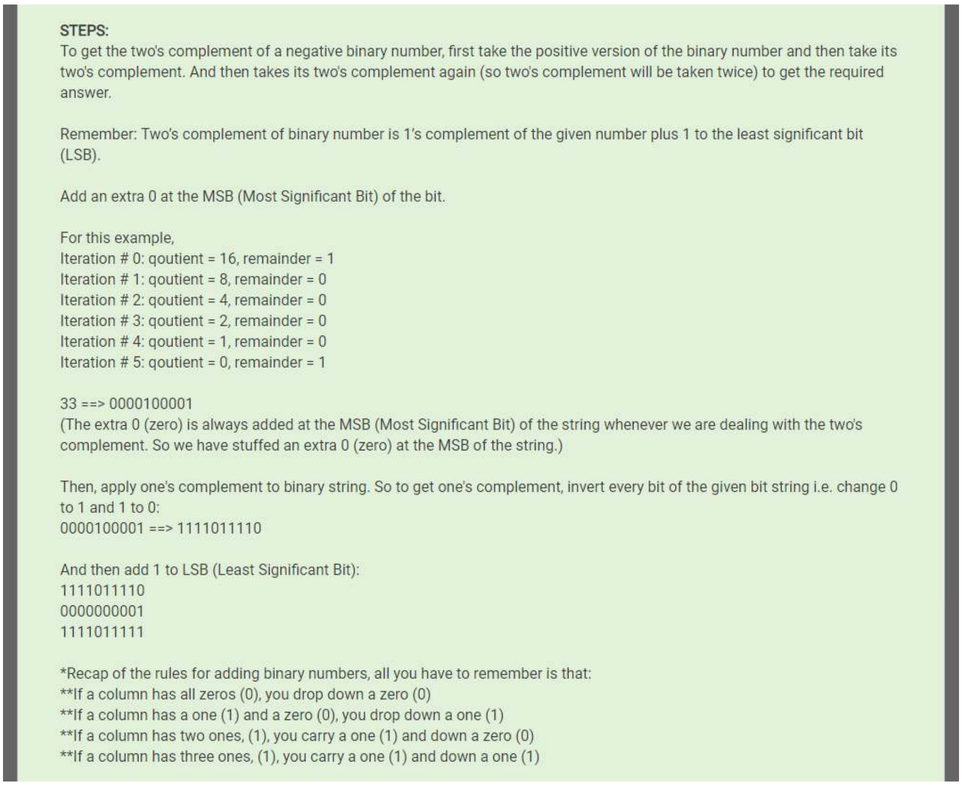


What is the two's complement of -33 in 10bit?

Base 2 (0000100001), Base 10 (33)

Figure 4.35: Test Case 17 Subcase 2: Chatbot's response for signed decimal number's two's complement in user-specified bits

Chatbot's response for signed decimal number's two's complement in user-specified bits is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
To get the two's complement of a negative binary number, first take the positive version of the binary number and then take its two's complement. And then takes its two's complement again (so two's complement will be taken twice) to get the required answer.

Remember: Two's complement of binary number is 1's complement of the given number plus 1 to the least significant bit (LSB).

Add an extra 0 at the MSB (Most Significant Bit) of the bit.

For this example,
Iteration # 0: qoutient = 16, remainder = 1
Iteration # 1: qoutient = 8, remainder = 0
Iteration # 2: qoutient = 4, remainder = 0
Iteration # 3: qoutient = 2, remainder = 0
Iteration # 4: qoutient = 1, remainder = 0
Iteration # 5: qoutient = 0, remainder = 1

33 ==> 0000100001
(The extra 0 (zero) is always added at the MSB (Most Significant Bit) of the string whenever we are dealing with the two's complement. So we have stuffed an extra 0 (zero) at the MSB of the string.)

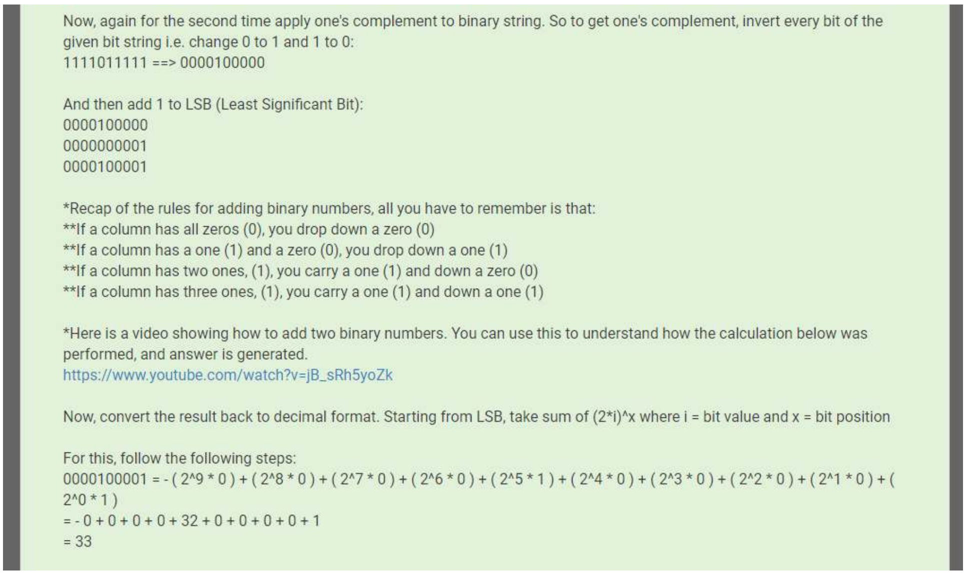
Then, apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
0000100001 ==> 1111011110

And then add 1 to LSB (Least Significant Bit):
1111011110
0000000001
1111011111

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)



Now, again for the second time apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
1111011111 ==> 0000100000

And then add 1 to LSB (Least Significant Bit):
0000100000
0000000001
0000100001

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Now, convert the result back to decimal format. Starting from LSB, take sum of (2*i)^x where i = bit value and x = bit position

For this, follow the following steps:
0000100001 = - (2^9 * 0) + (2^8 * 0) + (2^7 * 0) + (2^6 * 0) + (2^5 * 1) + (2^4 * 0) + (2^3 * 0) + (2^2 * 0) + (2^1 * 0) + (2^0 * 1)
= - 0 + 0 + 0 + 0 + 32 + 0 + 0 + 0 + 0 + 1
= 33

Figure 4.36: Test Case 17 Subcase 2: Explanation box for unsigned decimal number's two's complement in user-specified bits

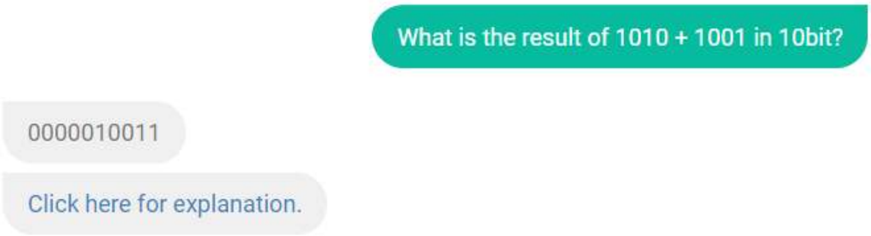
Subcase 3: Chatbot's response for Binary Addition in user-specified bits

Test Example String 1: 1010

Test Example String 2: 1001

Question: What is the result of 1010 + 1001 in 10bit?

Response:



What is the result of 1010 + 1001 in 10bit?

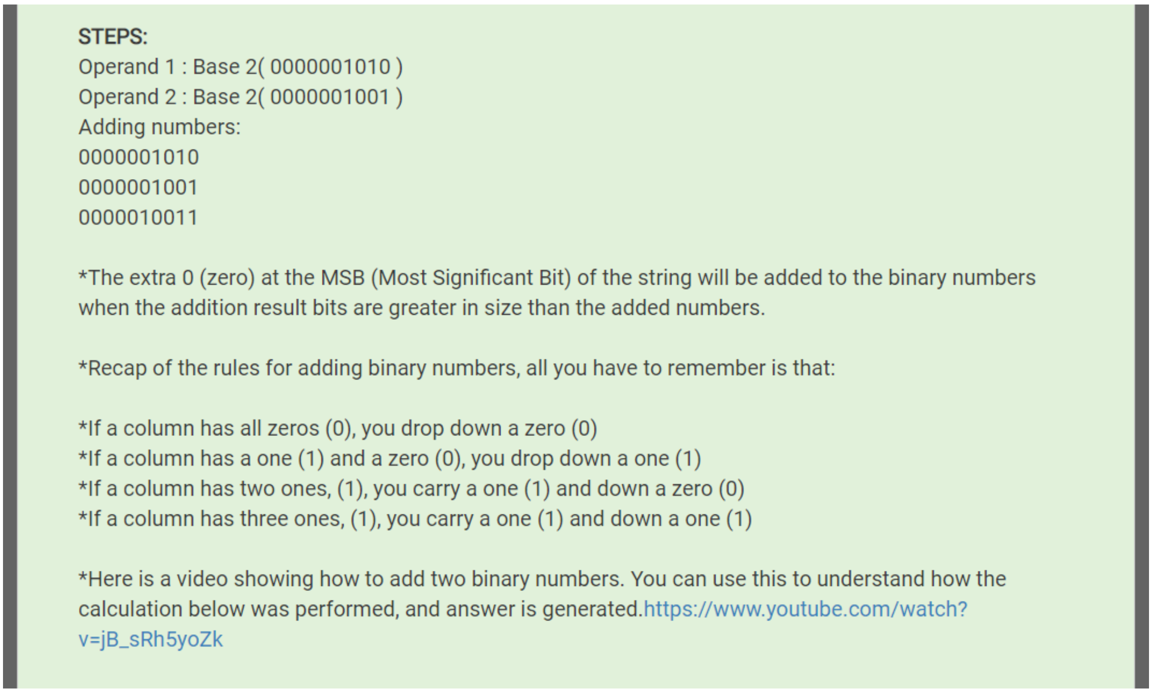
0000010011

Figure 4.37: Test Case 17 Subcase 3: Chatbot's response for Binary Addition in user-specified bits

Chatbot's response for Binary Addition in user-specified bits is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step

Downloaded by HASSAN ALI (hxn112233@gmail.com)

solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Operand 1 : Base 2(0000001010)
Operand 2 : Base 2(0000001001)
Adding numbers:
0000001010
0000001001
0000010011

*The extra 0 (zero) at the MSB (Most Significant Bit) of the string will be added to the binary numbers when the addition result bits are greater in size than the added numbers.

*Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.38: Test Case 17 Subcase 3: Explanation box for Binary Addition in user-specified bits

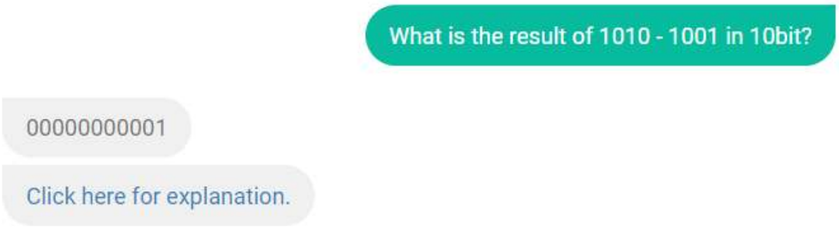
Subcase 4: Chatbot's response for Binary Subtraction in user-specified bits

Test Example String 1: 1010

Test Example String 2: 1001

Question: What is the result of 1010 - 1001 in 10bit?

Response:



What is the result of 1010 - 1001 in 10bit?

00000000001

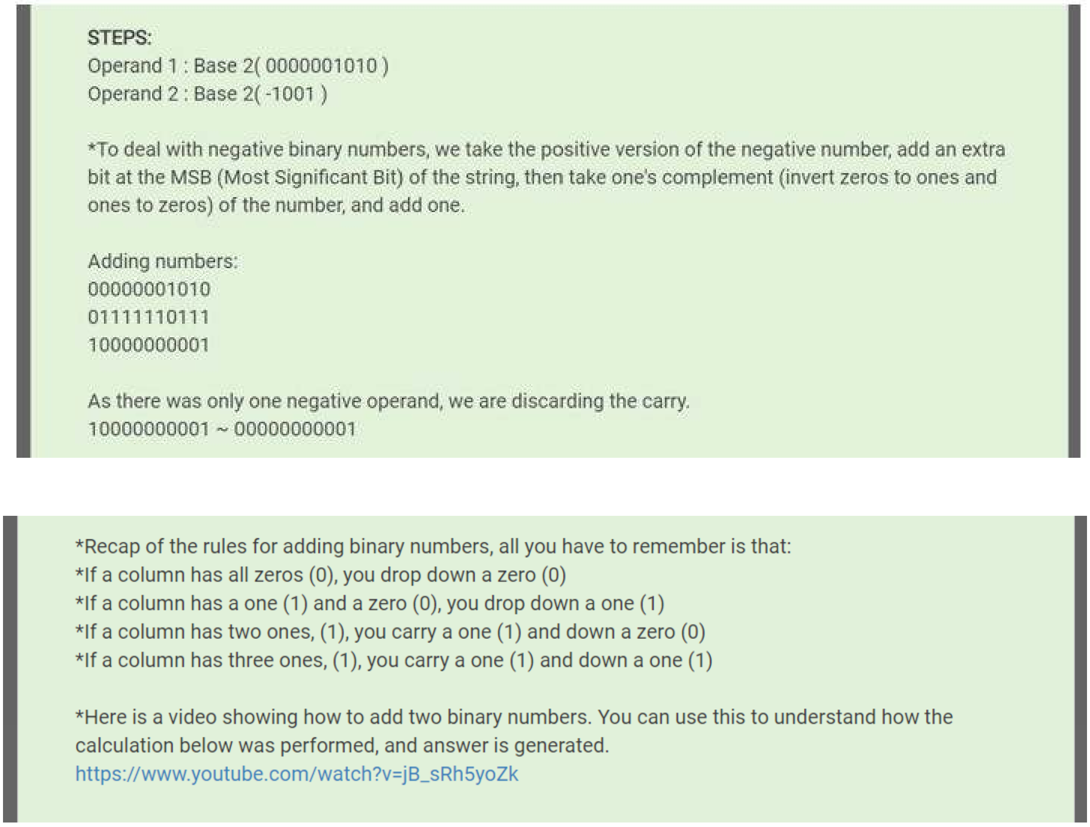
Click here for explanation.

Figure 4.39: Test Case 17 Subcase 4: Chatbot's response for Binary Subtraction in user-specified bits

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Chatbot’s response for Binary Subtraction in user-specified bits is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Operand 1 : Base 2(0000001010)
Operand 2 : Base 2(-1001)

*To deal with negative binary numbers, we take the positive version of the negative number, add an extra bit at the MSB (Most Significant Bit) of the string, then take one's complement (invert zeros to ones and ones to zeros) of the number, and add one.

Adding numbers:
00000001010
01111110111
10000000001

As there was only one negative operand, we are discarding the carry.
10000000001 ~ 00000000001

*Recap of the rules for adding binary numbers, all you have to remember is that:
*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.40: Test Case 17 Subcase 4: Explanation box for Binary Subtraction in user-specified bits

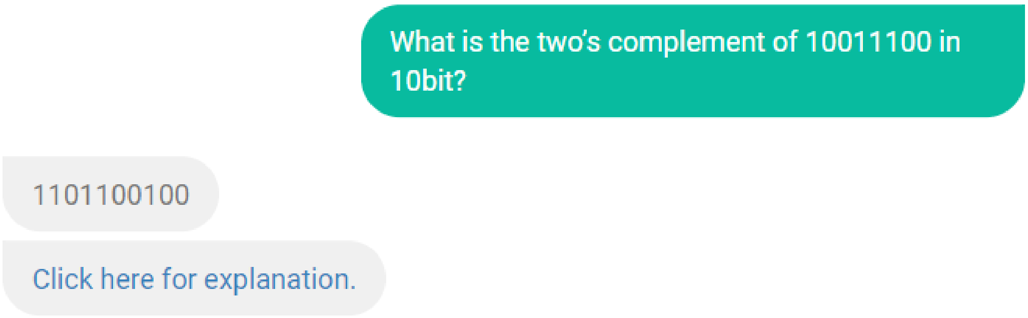
Subcase 5: Chatbot’s response for unsigned binary number’s two’s complement in user-specified bits

Test Example String: 10011100

Question: What is the two’s complement of 10011100 in 10bit?

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)



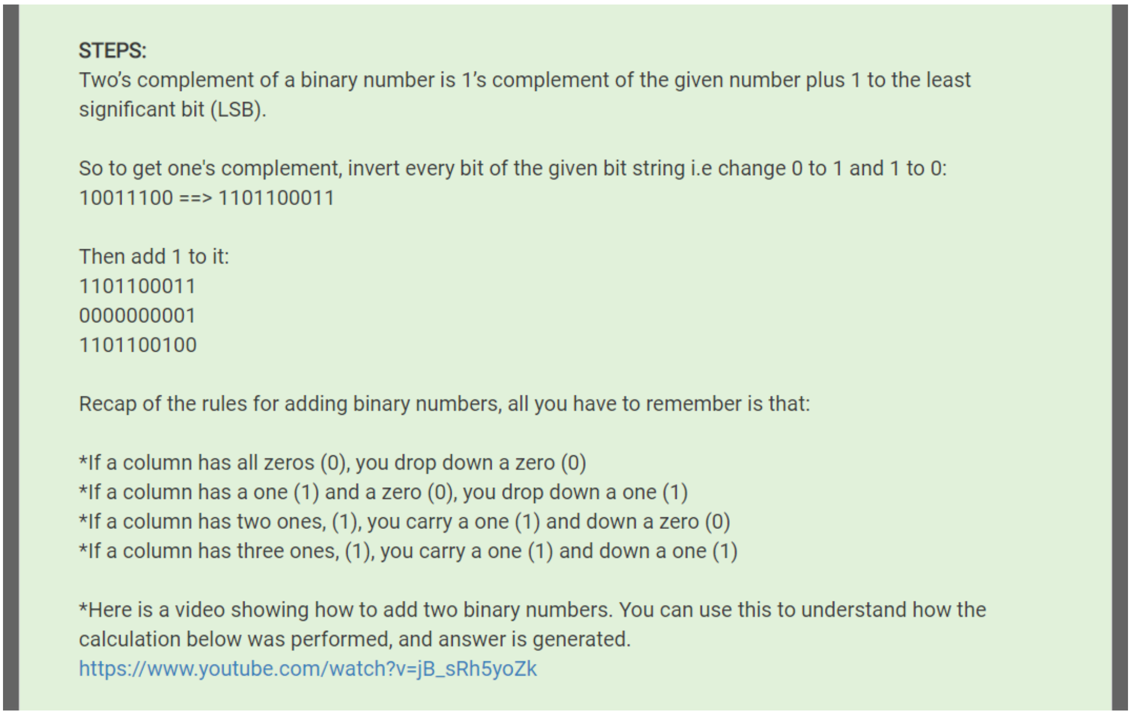
What is the two's complement of 10011100 in 10bit?

1101100100

Click here for explanation.

Figure 4.41: Test Case 17 Subcase 5: Chatbot's response for unsigned binary number's two's complement in user-specified bits

Chatbot's response for unsigned binary number's two's complement in user-specified bits is displayed on the main screen as shown above. By clicking on the 'Click Here for Explanation' link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.



STEPS:
Two's complement of a binary number is 1's complement of the given number plus 1 to the least significant bit (LSB).

So to get one's complement, invert every bit of the given bit string i.e change 0 to 1 and 1 to 0:
10011100 ==> 1101100011

Then add 1 to it:
1101100011
0000000001
1101100100

Recap of the rules for adding binary numbers, all you have to remember is that:

*If a column has all zeros (0), you drop down a zero (0)
*If a column has a one (1) and a zero (0), you drop down a one (1)
*If a column has two ones, (1), you carry a one (1) and down a zero (0)
*If a column has three ones, (1), you carry a one (1) and down a one (1)

*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.42: Test Case 17 Subcase 5: Explanation box for unsigned binary number's two's complement in user-specified bits

This document is available free of charge on studocu

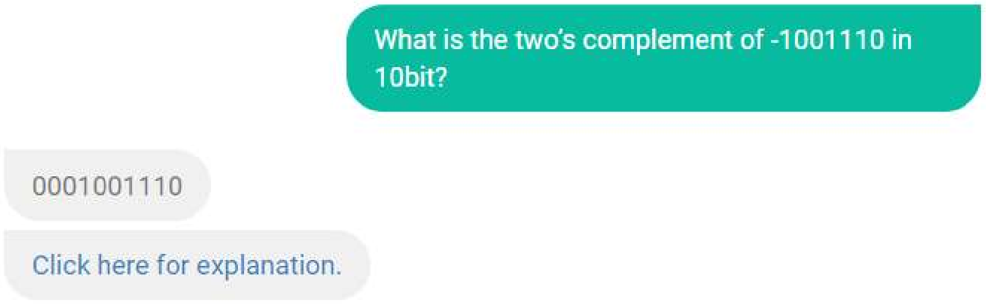
Downloaded by HASSAN ALI (hxn112233@gmail.com)

Subcase 6: Chatbot’s response for signed binary number’s two’s complement in user-specified bits

Test Example String: -1001110

Question: What is the two’s complement of -1001110 in 10bit?

Response:



What is the two's complement of -1001110 in 10bit?

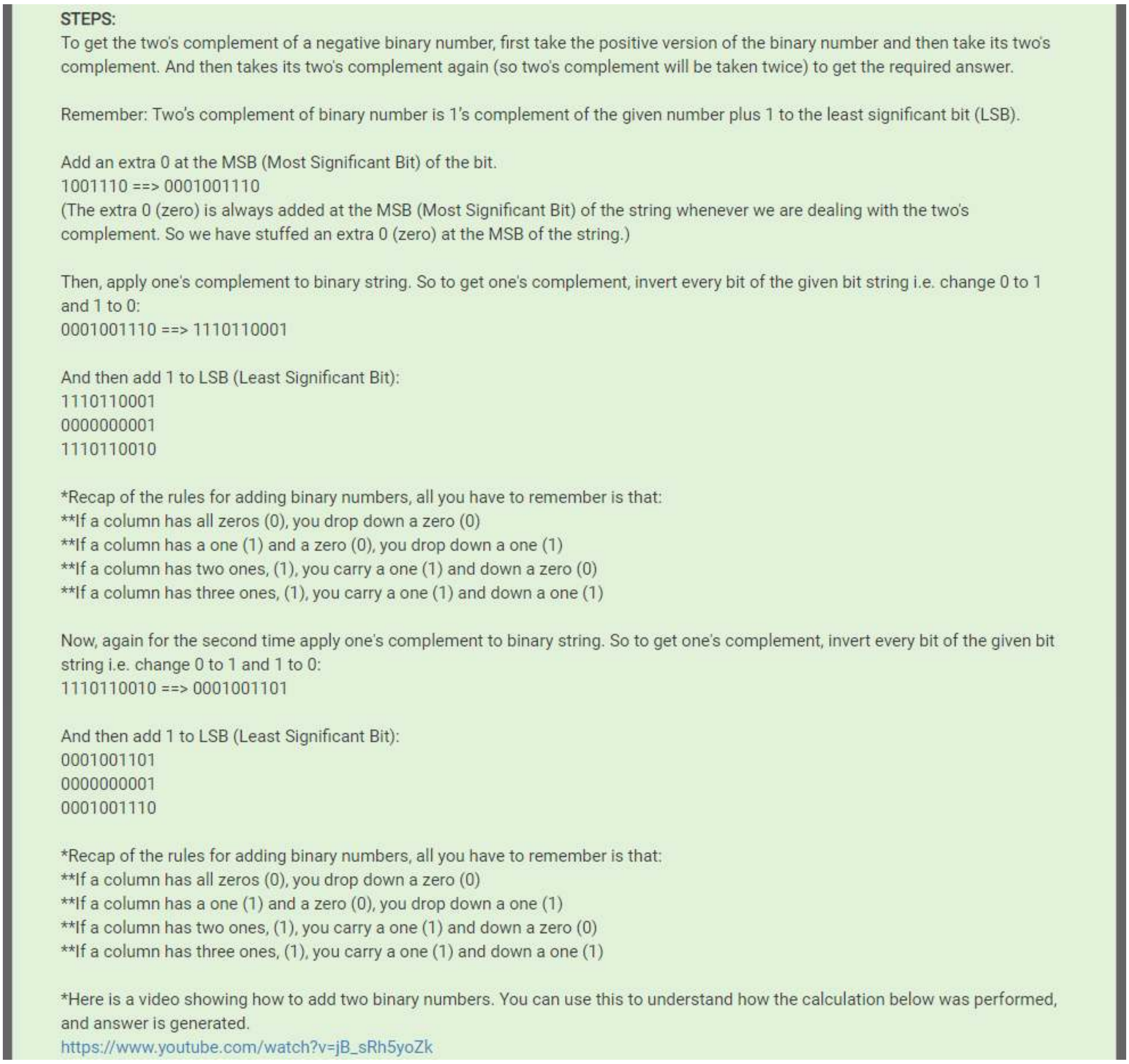
0001001110

Click here for explanation.

Figure 4.43: Test Case 17 Subcase 6: Chatbot’s response for signed binary number’s two’s complement in user-specified bits

Chatbot’s response for signed binary number’s two’s complement in user-specified bits is displayed on the main screen as shown above. By clicking on the ‘Click Here for Explanation’ link, an explanation box is generated with the step-by-step solution and explanation of the respective question along with a YouTube video link for further assistance as shown below.

Downloaded by HASSAN ALI (hxn112233@gmail.com)



STEPS:
To get the two's complement of a negative binary number, first take the positive version of the binary number and then take its two's complement. And then takes its two's complement again (so two's complement will be taken twice) to get the required answer.

Remember: Two's complement of binary number is 1's complement of the given number plus 1 to the least significant bit (LSB).

Add an extra 0 at the MSB (Most Significant Bit) of the bit.
1001110 ==> 0001001110
(The extra 0 (zero) is always added at the MSB (Most Significant Bit) of the string whenever we are dealing with the two's complement. So we have stuffed an extra 0 (zero) at the MSB of the string.)

Then, apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
0001001110 ==> 1110110001

And then add 1 to LSB (Least Significant Bit):
1110110001
0000000001
1110110010

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)

Now, again for the second time apply one's complement to binary string. So to get one's complement, invert every bit of the given bit string i.e. change 0 to 1 and 1 to 0:
1110110010 ==> 0001001101

And then add 1 to LSB (Least Significant Bit):
0001001101
0000000001
0001001110

*Recap of the rules for adding binary numbers, all you have to remember is that:
**If a column has all zeros (0), you drop down a zero (0)
**If a column has a one (1) and a zero (0), you drop down a one (1)
**If a column has two ones, (1), you carry a one (1) and down a zero (0)
**If a column has three ones, (1), you carry a one (1) and down a one (1)

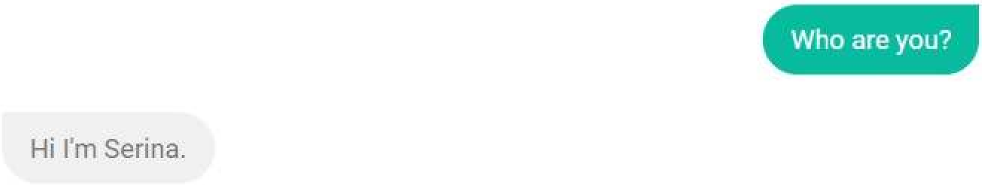
*Here is a video showing how to add two binary numbers. You can use this to understand how the calculation below was performed, and answer is generated.
https://www.youtube.com/watch?v=jB_sRh5yoZk

Figure 4.44: Test Case 17 Subcase 6: Explanation box for signed binary number's two's complement in user-specified bits

General information about the university

Question: Who are you?

Response:



Who are you?

Hi I'm Serina.

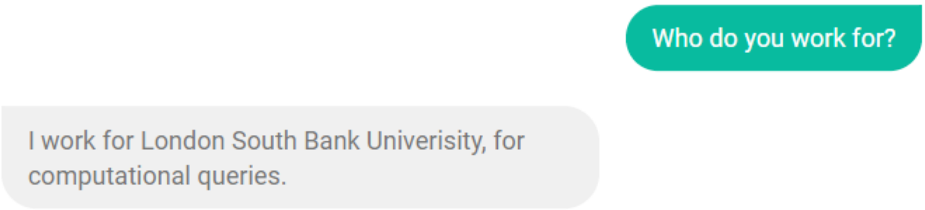
Figure 4.45: General information about the university Case 1

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Question: Who do you work for?

Response:



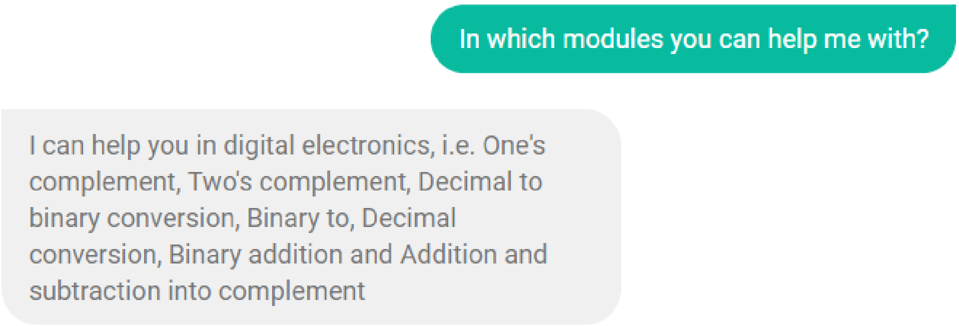
Who do you work for?

I work for London South Bank Univerisity, for computational queries.

Figure 4.46: General information about the university Case 2

Question: In which modules you can help me with?

Response:



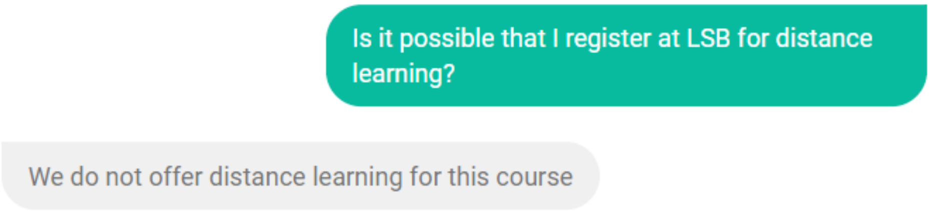
In which modules you can help me with?

I can help you in digital electronics, i.e. One's complement, Two's complement, Decimal to binary conversion, Binary to, Decimal conversion, Binary addition and Addition and subtraction into complement

Figure 4.47: General information about the university Case 3

Question: Is it possible that I register at LSB for distance learning?

Response:



Is it possible that I register at LSB for distance learning?

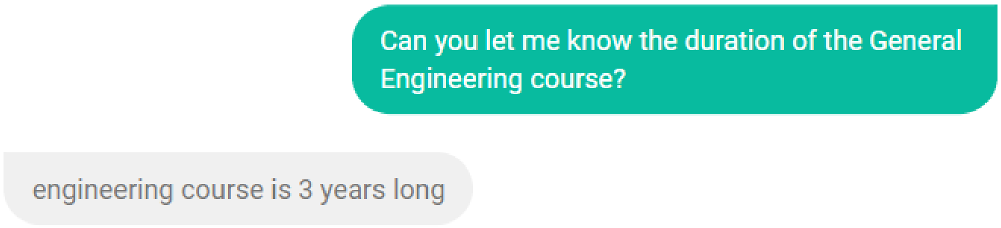
We do not offer distance learning for this course

Figure 4.48: General information about the university Case 4

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Question: Can you let me know the duration of the General Engineering course?

Response:



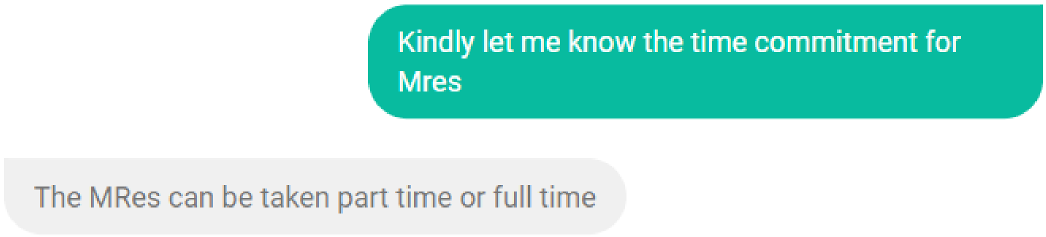
Can you let me know the duration of the General Engineering course?

engineering course is 3 years long

Figure 4.49: General information about the university Case 5

Question: Kindly let me know the time commitment for MRes.

Response:



Kindly let me know the time commitment for Mres

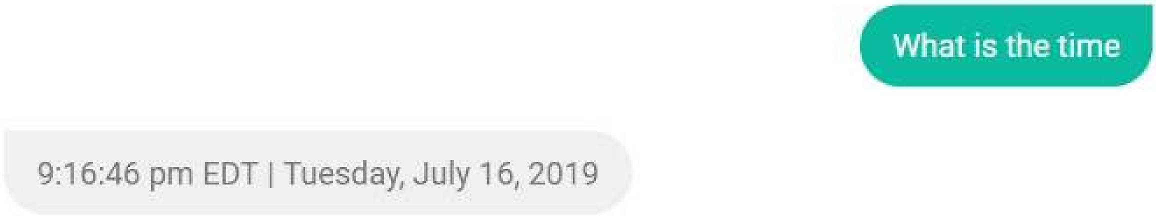
The MRes can be taken part time or full time

Figure 4.50: General information about the university Case 6

Generic Questions/Answers

Question: What is the time?

Response:



What is the time

9:16:46 pm EDT | Tuesday, July 16, 2019

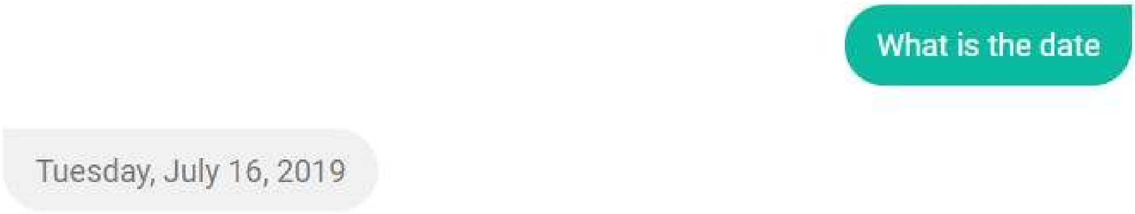
Figure 4.51: Generic Questions/Answers Testing Case 1

Question: What is the date?

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Response:



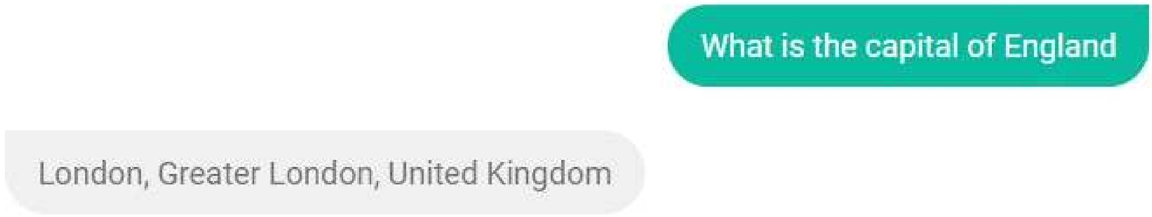
What is the date

Tuesday, July 16, 2019

Figure 4.52: Generic Questions/Answers Testing Case 2

Question: What is the capital of England?

Response:



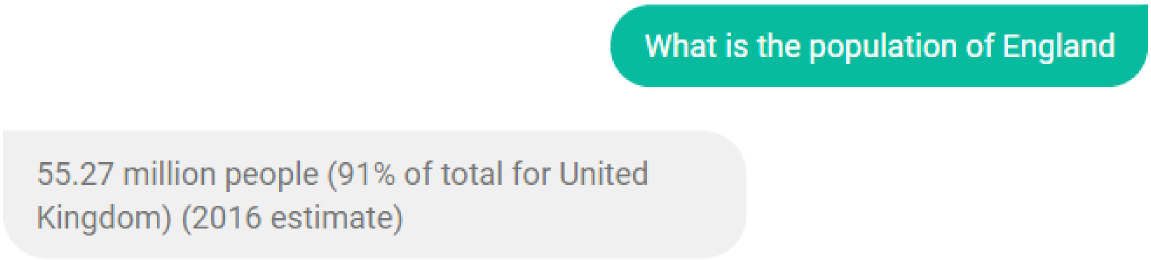
What is the capital of England

London, Greater London, United Kingdom

Figure 4.53: Generic Questions/Answers Testing Case 3

Question: What is the population of England?

Response:



What is the population of England

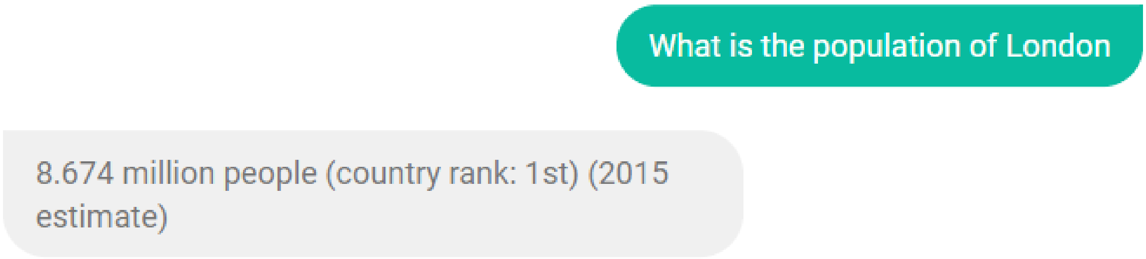
55.27 million people (91% of total for United Kingdom) (2016 estimate)

Figure 4.54: Generic Questions/Answers Testing Case 4

Question: What is the population of London?

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)



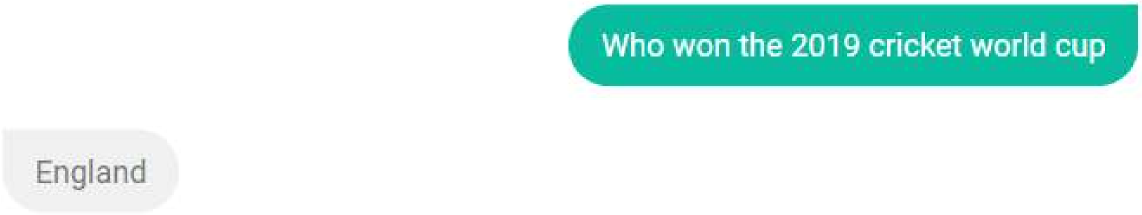
What is the population of London

8.674 million people (country rank: 1st) (2015 estimate)

Figure 4.55: Generic Questions/Answers Testing Case 5

Question: Who won the 2019 cricket world cup?

Response:



Who won the 2019 cricket world cup

England

Figure 4.56: Generic Questions/Answers Testing Case 6

Evaluation of Results

After testing the developed framework for all possible use cases, we have anaysed that the expected response and the obtained response are same. All the outcomes from the test cases lies under pass category.

The table below summarizes the outcomes of test cases of developed framework.

Test Case #	Description	Response Expected	Response Obtained	Outcome
1	Binary Addition	Provide accurate response and explanation	Accurate response and explanation provided	PASS

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

2	Binary Subtraction	Provide accurate response and explanation	Accurate response and explanation provided	PASS
3	Binary to Decimal Conversion	Provide accurate response and explanation	Accurate response and explanation provided	PASS
4	Decimal to Binary Conversion	Provide accurate response and explanation	Accurate response and explanation provided	PASS
5	One’s complement of unsigned decimal	Provide accurate response and explanation	Accurate response and explanation provided	PASS
6	One’s complement of signed decimal	Provide accurate response and explanation	Accurate response and explanation provided	PASS
7	One’s complement of unsigned binary	Provide accurate response and explanation	Accurate response and explanation provided	PASS
8	One’s complement of signed binary	Provide accurate response and explanation	Accurate response and explanation provided	PASS
9	Two’s complement of unsigned decimal	Provide accurate response and explanation	Accurate response and explanation provided	PASS

Downloaded by HASSAN ALI (hxn112233@gmail.com)

10	Two’s complement of signed decimal	Provide accurate response and explanation	Accurate response and explanation provided	PASS
11	Two’s complement of unsigned binary	Provide accurate response and explanation	Accurate response and explanation provided	PASS
12	Two’s complement of signed binary	Provide accurate response and explanation	Accurate response and explanation provided	PASS
13	Decimal Addition	Provide accurate response and explanation	Accurate response and explanation provided	PASS
14	Decimal Subtraction	Provide accurate response and explanation	Accurate response and explanation provided	PASS
15	Number of bits for Binary Representation of unsigned decimal	Provide accurate response and explanation	Accurate response and explanation provided	PASS
16	Number of bits for Binary Representation of signed decimal	Provide accurate response and explanation	Accurate response and explanation provided	PASS
17	Binary operations in user-specified bits	Provide accurate response and explanation	Accurate response and explanation provided	PASS
18	General Information about the university	Provide accurate response	Accurate response provided	PASS

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

19	Generic Questions/Answers	Provide accurate response	Accurate response provided	PASS

Table 4.1: Evaluation of Results

We can analyse from the table above that the developed framework have met all the functional requirements and all the test cases have provided accurate responses and explanation. Moreover, the explanation from the chatbot includes a YouTube video tutorial link where required. All the test cases outcomes fell under PASS category.

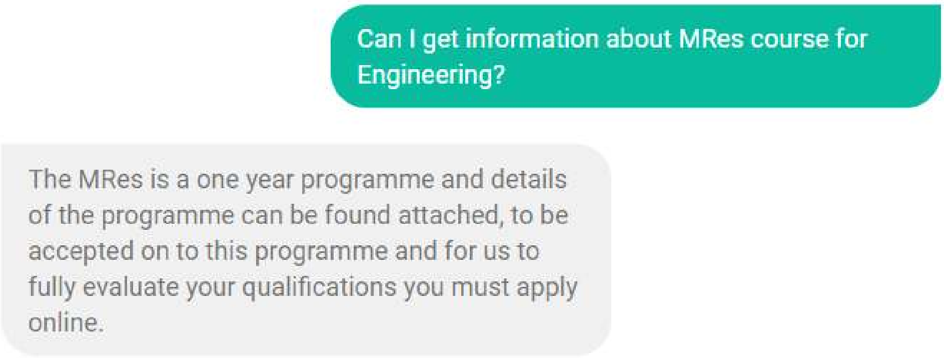
Evaluation on the basis of question type

There are broadly two categories in which the questions asked by the user is divided, and responses is collected and analysed. It is either a relevant question or an irrelevant question. The chatbot framework answers these questions on the basis of developed NLP. When the chatbot answer a user's question correctly, it implies that it was a valid question and the designed algorithm retrieved the answer from the framework. Conversely, if the chatbot is not able to answer a user's question justly, it indicates that it was an irrelevant question although valid i.e. outside the domain of the developed functional and non-functional requirements. Forward that the created algorithm was competent in retrieving the answer from the framework.

A suited question could be the user asking information about the MRes course for Engineering from the chatbot.

Response:

Downloaded by HASSAN ALI (hxn112233@gmail.com)



Can I get information about MRes course for Engineering?

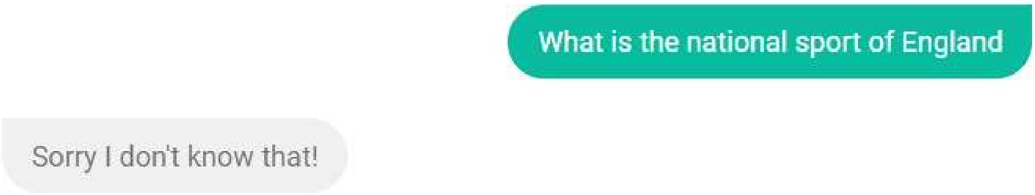
The MRes is a one year programme and details of the programme can be found attached, to be accepted on to this programme and for us to fully evaluate your qualifications you must apply online.

Figure 4.57: Chatbot Evaluation: Relevant Question

The reply was correct: “The MRes is a one-year programme and details of the programme can be found attached, to be accepted on to this programme and for us to fully evaluate your qualifications you must apply online.”

An unrelated type question asked, might be the user asking for the national sport of England from the chatbot.

Response:



What is the national sport of England

Sorry I don't know that!

Figure 4.58: Chatbot Evaluation: Irrelevant Question

The reply was not useful: “Sorry I don't know that!”

Evaluation by comparison with existing models

We will compare the developed framework with exisiting available models on two parameters i.e. classification according to educational intentionality vs classification according to tasks.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Classification according to Educational Intentionality

Depending on their nature, chatbots in education sector can be classified into two categories i.e. chatbots that do not have an educational intentionality and others that do. Chatbots that are incorporated into teaching tasks of an administrative nature including student guidance and personal assistance and of a support nature i.e. FAQs do not have educational intentionality.

Chatbots designed to foster teaching and learning directly have eduational intentionality. They can be further classified into two categories.

1. Chatbots that provide scaffolding for the learning process can adapt, select and sequence contents according to the student's needs and pace, aid reflection and metacognition processes and provide learning motivation.
2. Chatbots that provide exercise and practice programs for skills acquisition. These chatbots present a stimulus in the form of a question or problem, and the student gives an answer. This is automatically assessed by the chatbot, which gives immediate feedback to the student.

Classification according to tasks

Chatbots can be classified on the basis of tasks which includes administrative and management tasks to foster personal productivity, taking care of FAQs, student mentoring, motivation, practice of specific skills and abilities, simulations, reflection and metacognitive strategies and student learning assessment.

Downloaded by HASSAN ALI (hxn112233@gmail.com)

The table below summarizes the comparison between the developed framework and the other existing models according to the tasks they carry out and their educational intentionality.

		According to intentionality			
		Without Educational Intentionality		**With Educational Intentionality**	
		Academic guidance and personal agent	**Support (FAQs)**	**Tutor-support**	**Exercise and Practice**
According to tasks	**Administrative and management tasks to foster personal productivity**	Genie Hubert Ivy Pounce CourseQ Differ MOOCBuddy	Genie Ivy Pounce Otto	Genie	
	Answering administrative queries (FAQs)	Genie Ivy Pounce CourseQ Bot CEU Cardenal Herrera Serina	Genie Pounce Jill Watson Bot CEU Cardenal Herrera Serina	Serina	
	Mentoring	Serina	Jill Watson Serina	Pounce Jill Watson Differ Ani Botter Serina	Duolingo Pepper NAO
	Motivation			Differ Ani Botter	Duolingo
	Conversation for language learning			Ani	Duolingo
	Reflection and metacognitive strategies			The Guardian of History Replika	
	Student learning assessment	Serina		The Guardian of History Ani Serina	Pepper NAO

Table 4.2: Evaluation by comparison with existing models

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

CHAPTER 5

DISCUSSION

This chapter comprises of the analysis of the research objectives after the development of the chatbot framework to analyse whether the research objectives have been met or not, pros and cons of the implementation of the chatbot framework along with the discussion of the questionnaire survey[7] results.

Research Objectives Analysis

In Chapter 1, the research objectives were discussed including qualitative data, personalized learning, to establish a stronger teacher-student relationship, anytime assistance and customized teaching. We discussed earlier that it is very difficult for academic teachers to be aware of queries of the non-vocal students. So, teachers would be able to access the database of the developed chatbot framework and chat history and can point out the areas they may want to intensify more in class for specific students. Moreover, since the teachers would be capable of accessing the chat history and can know if the framework is answering the queries correctly, so the teacher can reach out to students individually to provide the proper approach to solve the questions, in case there is some ongoing error in the framework. Furthermore, we discussed under personalized learning that since students have different skills, interests, and capabilities when compared to one another, so the need for one-on-one lectures to the students can be fulfilled by the developed chatbot framework, since the teachers would have access to add separate learning modules to the

---

[7] Check Appendix A

Downloaded by HASSAN ALI (hxn112233@gmail.com)

framework as an administrator. Moreover, the developed framework will certainly establish a better teacher-student relationship since the framework will allow them to provide students with personal guidance and elaborate the curriculum with their own research perceptions. Furthermore, since the chatbot will be available on cloud 24/7, so the students would not have to wait anymore to have their queries attend to and can acquire the relevant knowledge they are counting on anytime. Finally, the developed chatbot framework will be able to transform the traditional teaching into customized teaching since the workload on the teachers will become less and the students will just have to access the specific module in the framework they are looking for. So, the students will be prevented from falling behind their peers in terms of learning and their grades.

Pros & Cons of the implementation of Chatbot framework

The pros and cons of the implementation of chatbot framework are listed below.

Pros

1. It will create a personalised learning environment for students
2. It will establish a better teacher-student relationship
3. It will provide 24/7 assistance to the students
4. It will promote the use of technology in academic institutions
5. It will improve query handling system for academic institutions
6. It can be used as a tool for Online Reputation Management.
7. It will promote innovative and customized teaching culture.

Cons

1. Since the developed framework is built using rule-based NLP approach, so it has a limited scope.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

2. The chatbot responses can be delayed if there’s an issue in the server side, so the students can be frustrated.

Questionnaire Survey and Results

The questionnaire survey results are collected from the period when the chatbot was operational and deployed. The responses were gathered from both undergraduate and postgraduate students of general engineering with a total of 60 students. The participats were approached and recruited through the business network of S.B. Pharma Group. Feedback returns were compiled from the students in a 14 days period of time. The questionnaire was designed in a Likert-scale manner where the participants can record their responses from a scale of Extremely Useful/Accurate to Not Useful/Accurate at all. The tables and pie charts of the persons surveyed recorded reactions, are included below.

Responses Recorded

Was the chatbot useful?

No. of Students	Extremely	Mostly	Moderately	Slightly	No
60	32	8	10	6	4

Table 5.1: Questionnaire Survey Results: Was the chatbot useful?

Figure 5.1: Pie Chart of Questionnaire Survey Results: Was the chatbot useful?

Were the responses accurate?

No. of Students	Extremely	Mostly	Moderately	Slightly	No
60	44	8	4	3	1

Table 5.2: Questionnaire Survey Results: Were the responses accurate?

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Figure 5.2: Pie Chart of Questionnaire Survey Results: Were the responses accurate?

Were the explanations accurate?

No. of Students	Extremely	Mostly	Moderately	Slightly	No
60	38	12	7	1	2

Table 5.3: Questionnaire Survey Results: Were the explanations accurate?

Figure 5.3: Pie Chart of Questionnaire Survey Results: Were the explanations accurate?

Would you like this to be the standard?

No. of Students	Strongly Agree	Somewhat Agree	Neutral	Somewhat Disagree	Strongly Disagree
60	42	7	2	3	6

Table 5.4: Questionnaire Survey Results: Would you like this to be the standard?

Figure 5.4: Pie Chart of Questionnaire Survey Results: Would you like this to be the standard?

Would you prefer to interact with the Chatbot via voice rather than text?

No. of Students	Strongly Agree	Somewhat Agree	Neutral	Somewhat Disagree	Strongly Disagree
60	28	15	5	6	6

Table 5.5: Questionnaire Survey Results: Would you prefer to interact with the Chatbot via voice rather than text?

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Figure 5.5: Pie Chart of Questionnaire Survey Results: Would you prefer to interact with the Chatbot via voice rather than text?

Survey Analysis

The responses from the questionnaire survey were amassed from both undergraduate and postgraduate students with a total of 60 students. In response to the question regarding the usefulness of the bot, 53% of the participants found the chatbot to be extremely useful. 13% found it to be mostly useful, 17% were of the opinion that it is moderately useful. 10% disclosed the chatbot to be slightly useful, whilst 7% of the participants said that the bot is not useful at all. 75% of the participants regarded the bot's responses as extremely accurate when they were asked whether the chatbot responses were accurate, 12% claimed the responses to be mostly accurate, 6% said that the responses are moderately accurate. In contrast only 5% and 2% of the participants stated the responses to be slightly accurate or not accurate at all respectively. Further to this, 63% of the participants regarded the bot's explanations towards asked questions as extremely accurate, when they were asked if the chatbot explanations were accurate. 20% remarked the explanations to be mostly accurate, 12% proclaimed that the explanations are moderately accurate. But 2% and 3% of the participants voiced that the explanations were slightly accurate or not accurate at all respectively. In reply to the question if the participants would like the chatbot to be the standard. 70% of the participants strongly agreed, 12% said that they are somewhat agreed, 3% remained neutral, 5% somewhat disagreed, and 10% of the participants strongly disagreed with the concept that the instituted chatbot should be the standard. Finally, the

Downloaded by HASSAN ALI (hxn112233@gmail.com)

participants were asked if they would prefer to interact with the chatbot via voice rather than text. 47% of them strongly agreed that bot should be voice based as well, 25% somewhat agreed, 8% stayed neutral, 10% somewhat disagreed and 10% strongly disagreed as well.

The survey results summary is listed in the tables below.

Was the chatbot useful?

Sample Size	**Extremely**	**Mostly**	**Moderately**	**Slightly**	**No**
60	53%	13%	17%	10%	7

Table 5.6: Questionnaire Survey Results Summary: Was the chatbot useful?

Were the responses accurate?

Sample Size	**Extremely**	**Mostly**	**Moderately**	**Slightly**	**No**
60	75%	12%	6%	5%	2%

Table 5.7: Questionnaire Survey Results Summary: Were the responses accurate?

Were the explanations accurate?

Sample Size	**Extremely**	**Mostly**	**Moderately**	**Slightly**	**Not**
60	63%	20%	12%	2%	3%

Table 5.8: Questionnaire Survey Results Summary: Were the explanations accurate?

Would you like this to be the standard?

Sample Size	**Strongly Agree**	**Somewhat Agree**	**Neutral**	**Somewhat Disagree**	**Strongly Disagree**

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

60	70%	12%	3%	5%	10%

Table 5.9: Questionnaire Survey Results Summary: Would you like this to be the standard?

Would you prefer to interact with the Chatbot via voice rather than text?

Sample Size	**Strongly Agree**	**Somewhat Agree**	**Neutral**	**Somewhat Disagree**	**Strongly Disagree**
60	47%	25%	8%	10%	10%

Table 5.10: Questionnaire Survey Results Summary: Would you prefer to interact with the Chatbot via voice rather than text?

CHAPTER 6

CONCLUSION

Summary

This project aimed to reduce the burden on teachers and stressed-out faculty, along with the notion of gaining an impeccable teacher-student relationship. The project also aspired to prepare students for the personalised learning, by creating a convincing chatbot which will assist students in their queries related to binary operations. The composed chatbot system is capable of handling vast binary operations and can help students in their day-to-day lessons. With this bot at work, students would not have to wait to get their queries addressed and obtain the information they are seeking. Thus, pupils can achieve personalised learning. Beyond that, it will help in building a strong teacher-student relationship, as the teachers will have the favourable circumstances to provide students with personal guidance and expand the curriculum with their own research insights. In this way, teachers can dispense impactful mentorship. Complementary to this, expending time on lengthy and complex binary calculations such as taking

Downloaded by HASSAN ALI (hxn112233@gmail.com)

two's complement. Or minor calculations for example base 2 and base 10 conversions, can be avoided employing the designed binary chatbot. As a consequence students can truly think creatively, and learn material without having to hunt for the calculations. As the time for this is now available.

Python Flask and MongoDB were used for the backend and HTML5, CSS3 and jQuery were used for the frontend of the framework. Deployment was exercised through Heroku, which allows developers to easily deploy, manage and scale their web applications on cloud. Not only that, MongoDB Atlas, NLTK, AWS Lambda Functions, AWS Lex, Boto3, Pymongo and Gunicorn were used as Third-Party Tools and Libraries. The chatbot's functional and non-functional requirements were analysed and a relevant methodology was devised. The questionnaire survey was conducted after the chatbot was operational, and their responses were recorded and the evaluation of results was generated. 53% of the participants remarked the chatbot to be extremely useful, 75% of the participants regarded the chatbot's responses as extremely accurate, 63% of the participants noted the chatbot's explanation towards asked questions as extremely accurate. 70% of the participants strongly agreed that the launched chatbot should be the standard, and 47% of them strongly agreed that chatbot should be voice-based as an extra characteristic.

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

CHAPTER 7

LIMITATIONS & RECOMMENDATIONS

Framework Limitations

The developed framework limitations are as follows.

Text-based Input

The developed framework supports only text-based input from the students or users.

Rule-based approach

The chatbot framework is developed using the rule-based approach in which fixed set of rules are defined for NLP algorithm to parse user input. So the developed framework has a limited scope.

Single Language Support

The developed framework supports English language only. So the framework would not be able to assist the user’s input in any other language.

Framework Testing Limitations

There were no specific limitations in the testing of the framework. MongodB and AWS Lex supports thousands of queries in their free version. So no

Downloaded by HASSAN ALI (hxn112233@gmail.com)

limitations were recorded in the testing phase. Moreover, the questionnaire survey participants also did not record any limitations during their survey.

Future Work

Based on the framework limitations listed above and the responses recorded from the survey participants, the recommendations for future work are listed below.

Machine Learning

All the questions and their answers given by the chatbot, are stored in a database on Atlas Instance. It can be used in future to implement machine learning and AI approach for the chatbot to provide more accurate answers for students.

Voice-based

In the questionnaire survey, it was asked to the participants if they wanted the bot to be voice-based rather than text-based. 47%[8] of those questioned strongly agreed that the chatbot should be voice-based as well. Hence the bot framework can be altered in future to support voice functionality additionally.

Advanced NLP

The framework can be altered to support advanced parsing techniques to achieve a better natural language processing in future.

---

[8] See Survey Analysis in Chapter 5

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Better Interface

Chatbot user interface can be further upgraded and boosted to reduce the development time of the framework. Moreover, android and iOS applications can be introduced for the students in the future. In addition to this, we can also add frequently asked questions (FAQS) related to the functionality of the modules.

Multiple Language Support

If the chatbot has the capability to support multiple languages, it will be deemed significantly more user-friendly and cooperative.

Downloaded by HASSAN ALI (hxn112233@gmail.com)

CHAPTER 8

RESEARCH ETHICS

In Chapter 3, it was indicated in the non-functional requirements that the research will follow the research ethics including that any kind of information about the user, questionnaire survey participants and teachers using the chatbot will not be stored or processed by any means.

The following research ethics have been followed during this research.

1. Informed consent have been obtained from potential research participants
2. The risk of harm to participants is minimized.
3. The anonymity and confidentiality of the research participants have been protected.
4. Deceptive research practices have been avoided at all costs.
5. The participants were given the right to withdraw from the reaearch.

Moreover, to ensure that the developed chatbot framework is GDPR compliant, the following practices have been used.

1. Since the chatbot is considered to be a data collecting and processing tool, so it falls under GDPR legislation and we are intended not to use the user's personal data for any use.
2. In case the university offers to extend the framework domain and the need for use of personal data comes up, then we will need to ask for explicit consent from users to use their personal data.
3. The users will be provided access and ability to download their saved information using a query and response format.
4. No personal data will be stored without any legitimate reason and only if the users have provided their explicit consent to do so.

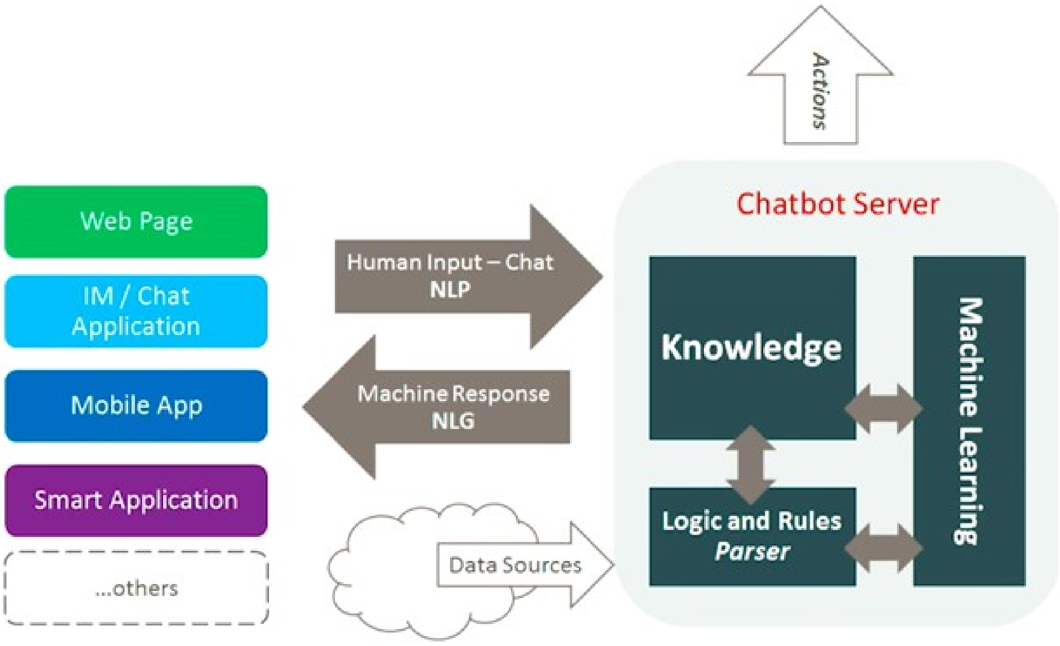
This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

5. The users will be allowed to delete and retrieve their data.
6. A privacy policy will be created and update for the users.

APPENDIX A. Anatomy of a self-learning chatbot

Downloaded by HASSAN ALI (hxn112233@gmail.com)



APPENDIX B. Questionnaire Survey

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Was the Chatbot useful?

Extremely Useful	Mostly Useful	Moderately Useful	Slightly Useful	Not Useful
☐	☐	☐	☐	☐

Were the responses accurate?

Extremely Accurate	Mostly Accurate	Moderately Accurate	Slightly Accurate	Not Accurate
☐	☐	☐	☐	☐

Were the explanations accurate?

Extremely Accurate	Mostly Accurate	Moderately Accurate	Slightly Accurate	Not Accurate
☐	☐	☐	☐	☐

Would you like this to be the standard?

Strongly Agree	Somewhat Agree	Neutral	Somewhat Disagree	Strongly Disagree
☐	☐	☐	☐	☐

Would you prefer to interact with the Chatbot via voice rather than text?

Strongly Agree	Somewhat Agree	Neutral	Somewhat Disagree	Strongly Disagree
☐	☐	☐	☐	☐

APPENDIX C. NLP

Tokenization

```
import nltk
import re, math, json
#nltk.download('punkt')
required_bits= 4
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
default_bits = 0
isSignednumber = False
userbits = 0
keyword_list = [
    'write', 'one', 'two','ones',
    'twos', 'binary',
    "one's", "two's", 'compliment',
    'convert', 'complement', 'represent',
    'number', 'bits', 'value',
    'decimal', '+', '-',
    'sum', 'difference', 'required'
]
# retrieve common keywords.
def compare(query):
    tokenize = nltk.word_tokenize(query)
    return [x for x in tokenize if x in keyword_list]
```

Function 01: Input String Format Checking

```
def check_binary_format(string) :
    p = set(string)
    s = {'0', '1'} # declare set of '0', '1' .

    # case 1: string has both 1s and 0s
    # case 2: string only has 1s
    # case 3: string only has 0s
    if s == p or p == {'0'} or p == {'1'}:
        return "yes"
    else:
        return "no"
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Function 02: B2D Conversion

```
    if len(binary_numbers) == 1:
        steps = 'Starting from LSB, take sum of (2*i)^x where i = bit value and
x = bit position.<br><br>'
        steps = steps + ' For this, follow the following steps: <br>' +
binary_numbers[0] + ' = '
        sum = 0
        position = len(binary_numbers[0])
            sum = sum + (2 * int(binary_numbers[0][i], base = 2) ** position)
            position = position - 1
            steps = steps + '(2*' + binary_numbers[0][i] + ')^' + str(position) + '
+ '

        steps = steps[0: len(steps) - 2]
        steps = steps + ' = ' + str(int(binary_numbers[0], base = 2))
        steps = steps + '<br>'
        return [str(int(binary_numbers[0], base = 2)), steps[:len(steps) - 4]]

    elif len(decimal_numbers) == 1:
        return decimal_to_binary(decimal_numbers)

    else:
        return ['Error! 1 argument required, given 0 or more than 1', ""]
```

Function 03: One's Complement of a Signed Integer

```
def signed_ones_complement(decimal_numbers, all_numbers):
    global required_bits
    i = all_numbers[0]
    result = []
    answer = ""
    steps = ""
    if len(all_numbers) == 1:
        if(all_numbers[0][1] == "int"):
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        steps = steps + 'First we convert the positive version of decimal number to its binary form:<br> For this Example, <br>'
        num = all_numbers[0][0]
        if(num > 0):
            count = 0
            carry = 0
            bits = math.ceil(math.log(int(num), 2))
            while (count < bits):
                steps = steps + "Iteration # " + str(count) + ": " + "remainder = " + str(int(num/2))
                carry = num % 2
                num = num / 2
                steps = steps + ', carry = ' + str(int(carry)) + '<br>'
                count = count + 1
        answer = str(bin(all_numbers[0][0]).replace("0b", ""))
    else:
        answer = str(all_numbers[0][0])
    anslen = len(answer) + 1
    steps = steps + "<br>Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :<br>"
    r_bit = anslen
    one_comp = one_complement([answer])
    steps = steps + answer + " ==> " + one_comp[0] + "<br><br>Then, add one to it:<br>"
    add_one = "1"
    if(anslen < required_bits):
        r_bit = required_bits
        answer = answer.zfill(required_bits)
    else:
        answer = answer.zfill(anslen)
    result.append(getBinTwosComplement(answer,r_bit))
    steps = steps + result[0][1]
    answer = answer.zfill(len(result[0][0]))
    add_one = add_one.zfill(len(result[0][0]))

    steps = steps + answer + "<br>" + add_one + "<br>" + result[0][0]
    steps = steps + "<br><br>Recap of the rules for adding binary numbers, all you have to remember is that:<br><br>"\
    steps = steps + "<br><br>Invert every bit of the given bit string i.e change 0 to 1 and 1 to 0 :<br>"
    steps = steps + answer + " ==> " + temp
    if(all_numbers[0][1] == "int"):
        steps = steps + "<br>Lastly convert the answer to its decimal value using, take sum of (2*i)^x where i = bit value and x = bit position: <br>"
        answer = temp
        ans = answer
        answer=0
        steps = steps + ""
        next_step = "= "
        if(ans[0]=='1'):
            answer = int(math.pow(2, len(ans)-1)) * -1
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
                steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
                next_step = next_step + "- " + str(int(math.pow(2, len(ans)-1)))
            else:
                steps = steps + "- ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
                next_step = next_step + "- 0"
            index= len(ans) - 1
            index1= 1
            while (index > 1):
                steps = steps + "+ ( 2^" + str(index-1) + " * " + ans[index1]+" ) "
                if(ans[index1]=="1"):
                else:
                    next_step = next_step + " + 0"
                index1 = index1 + 1
                index = index - 1
            steps = steps + "+ ( 2^0"  + " * " + ans[len(ans)-1]+" )<br>"
            if(ans[len(ans)-1] == "1"):
                answer = answer + 1
                next_step = next_step + " + 1<br>"
            else:
                next_step = next_step + " + 0<br>"
            next_step = next_step + " = " + str(answer) + "<br>"
            steps = steps + next_step
            final_ans = "Base 2 ( " + ans + " ), Base 10 ( " + str(answer) + ")"
        else:
            final_ans = temp
    else:
        return ["Error! 1 argument required, given 0 or more than 1", ""]
    return [final_ans, steps]
```

Function 04: Two’s Complement of a Signed Integer

```
def signed_twos_complement(decimal_numbers, all_numbers):
    global required_bits
    i = all_numbers[0]
    result = []
    answer = ""
    if len(all_numbers) == 1:
        if(all_numbers[0][1] == "int"):
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            steps = steps + '<br> For this example, <br>'
            num = all_numbers[0][0]
            if(num > 0):
                count = 0
                carry = 0
                bits = math.ceil(math.log(int(num), 2))
                while (count < bits):
                    steps = steps + "Iteration # " + str(count) + ": " + "qoutient = "
+ str(int(num/2))
                    carry = num % 2
                    num = num / 2
                    steps = steps + ', remainder = ' + str(int(carry)) + '<br>'
                    count = count + 1
                steps = steps + "<br>"
            answer = str(bin(all_numbers[0][0]).replace("0b", ""))
            anslen = len(answer) + 1
            r_bit = anslen
            if(anslen < required_bits):
                r_bit = required_bits
                answer = answer.zfill(required_bits)
            else:
                answer = answer.zfill(anslen)
        else:
            answer = all_numbers[0][0]
            anslen = len(answer) + 1

            r_bit = anslen
            if(anslen < required_bits):
                r_bit = required_bits
                answer = answer.zfill(required_bits)
            else:
                answer = answer.zfill(anslen)
        steps = steps + str(all_numbers[0][0]) + " ==> " + answer +"<br>(The
extra 0 (zero) is always added at the MSB (Most Significant Bit) of the
string whenever we are dealing with the two's complement. So we have
stuffed an extra 0 (zero) at the MSB of the string.)<br><br>"
        one_comp = one_complement([answer])

        result.append(getBinTwosComplement(answer,r_bit))
        steps = steps + "Then, apply one's complement to binary string. So to
get one's complement, invert every bit of the given bit string i.e. change 0
to 1 and 1 to 0:<br>"
        steps = steps + answer + " ==> " + one_comp[0] +"<br><br>And then
add 1 to LSB (Least Significant Bit):<br>"
        r_bit = len(result[0][0])
        add_one = "1"
        add_one = add_one.zfill(len(answer))
        steps = steps + one_comp[0] + " <br> " + add_one + "<br>" +
result[0][0] + "<br>"\
        steps = steps + "<br><br>Now, again for the second time apply one's
complement to binary string. So to get one's complement, invert every bit
of the given bit string i.e. change 0 to 1 and 1 to 0:<br>"
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        one_comp = one_complement([result[0][0]])
        steps = steps + result[0][0] + " ==> " + one_comp[0] + "<br>"
        steps = steps + "<br>And then add 1 to LSB (Least Significant
Bit):<br>"
        new_ans = str(bin(int(one_comp[0], base = 2) + 1).replace('0b',
'')).zfill(r_bit)
        add_one = add_one.zfill(len(new_ans))
        steps = steps + one_comp[0] + " <br> " + add_one + "<br>" +
new_ans + "<br>"\
            "<br>*Recap of the rules for adding binary numbers, all you have
to remember is that:"\
        if all_numbers[0][1] == "int":
            steps = steps + "<br> <br>Now, convert the result back to decimal
format. Starting from LSB, take sum of (2*i)^x where i = bit value and x =
bit position<br><br>For this, follow the following steps: <br>"
        a = result[0][0]
        reqBit = len(a)
        temp = ""
        answer = str(bin(int(temp, base = 2) + 1).replace('0b', '')).zfill(r_bit)
        if(len(answer) > reqBit):
            answer = answer[1:]
        if(all_numbers[0][1] == "int"):
            ans = answer
            steps = steps + ans + " = "
            next_step = "= "
            answer = 0
            if(ans[0]=='1'):
                answer = int(math.pow(2, len(ans)-1)) * -1
                steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
                next_step = next_step + "- " + str(int(math.pow(2, len(ans)-1)))
            else:
                steps = steps + "- ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
                next_step = next_step + "- 0"
            index= len(ans) - 1
            index1= 1
            while (index > 1):
                steps = steps + "+ ( 2^" + str(index-1) + " * " + ans[index1]+" ) "
                if(ans[index1]=="1"):
                else:
                    next_step = next_step + " + 0"
                index1 = index1 + 1
                index = index - 1
            steps = steps + "+ ( 2^0"  + " * " + ans[len(ans)-1]+" )<br>"
            if(ans[len(ans)-1] == "1"):
                answer = answer + 1
                next_step = next_step + " + 1<br>"
            else:
                next_step = next_step + " + 0<br>"
            next_step = next_step + " = " + str(answer) + "<br>"
            steps = steps + next_step
            final_ans = "Base 2 ( " + ans + " ), Base 10 ( " + str(answer) + ")"
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        else:
            final_ans = answer
    else:
        return ["Error! 1 argument required, given 0 or more than 1", ""]
    return [final_ans, steps]
```

Function 05: D2B Conversion

```
    if len(decimal_numbers) == 1:
        steps = steps + 'remainder becomes smaller than 2. Then read all the
carry in backward (bottom to top) direction.<br>'
        steps = steps + ' For this example:<br>'
        num = decimal_numbers[0]
        if(num > 0):
            count = 0
            carry = 0
            bits = math.ceil(math.log(int(num), 2))
            while (count < bits):
                steps = steps + "Iteration # " + str(count) + ": " + "quotient = " +
str(int(num/2))
                carry = num % 2
                num = num / 2
                steps = steps + ', remainder = ' + str(int(carry)) + '<br>'
                count = count + 1
            steps = steps + "<br>" + str(decimal_numbers[0]) + " ==> " +
str(bin(decimal_numbers[0]).replace("0b", ""))
            print("till mod ", int(decimal_numbers[0])%2, " " ,
decimal_numbers[0])
            x= 2
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            num1 = int(decimal_numbers[0])
            temp = False
            while(x < num1):
                if x*2 == num1:
                    temp = True
                x = x * 2
            if(num1 == 2):
                temp = True
            if(temp):
                print("is mod")
                steps = steps + "<br><br>(*Note: the 1 is used in the answer
because the final iteration is 2 divided by 2 which equals 1)"
        else:
            steps = "<b> its a signed number.</b>"
        print(steps)
        answer = str(bin(decimal_numbers[0]).replace("0b", ""))
        return [answer, steps]
    else:
        return ["Error! 1 argument required, given 0 or more than 1", ""]
```

Function 06: Signed Decimal to Binary Conversion

```
def signed_decimal_to_binary(decimal_numbers, all_numbers):
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
    global required_bits
    result = []
    steps = ""
    if len(decimal_numbers) == 1:
        steps = steps + 'For a negative number: The first step is to convert
the positive version of the given negative number into binary. To perform
this, divide the number by 2 repeatedly until remainder becomes smaller
than 2. Then read all the carry in backward (from bottom to top)
direction.<br><br>'
        steps = steps + ' For this example, <br>'
        num = decimal_numbers[0]
        if(num > 0):
            count = 0
            carry = 0
            bits = math.ceil(math.log(int(num), 2))
            while (count < bits):
                steps = steps + "Iteration # " + str(count) + ": " + "quotient = " +
str(int(num/2))
                carry = num % 2
                num = num / 2
                steps = steps + ', remainder = ' + str(int(carry)) + '<br>'
                count = count + 1
            steps = steps + "<br>"
        answer = str(bin(decimal_numbers[0]).replace("0b", ""))
        anslen = len(answer) + 1
        r_bit = anslen
        if(anslen < required_bits):
            r_bit = required_bits
            answer = answer.zfill(required_bits)
        else:
            answer = answer.zfill(anslen)
        steps = steps + str(decimal_numbers[0]) + " ==> " + answer
        x= 2
        num1 = int(decimal_numbers[0])
        temp = False
        while(x < num1):
            if (x*2 * 1) == num1:
                temp = True
            x = x * 2
        if(num1 == 2):
            temp = True
        if(temp):
            steps = steps + "<br><br>(*Note: the 1 is used in the answer
because the final iteration is 2 divided by 2 which equals 1)<br>"
        steps = steps + "<br>(The extra 0 (zero) is always added at the MSB
(Most Significant Bit) of the string whenever we are dealing with the two's
complement. So we have stuffed an extra 0 (zero) at the MSB of the
string.<br>"
        result.append(getBinTwosComplement(answer,r_bit))
        ones_complement = one_complement([answer])
        steps = steps + answer + " ==> " + ones_complement[0]
        add_one="1"
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        add_one = add_one.zfill(len(result[0][0]))
        ones_complement[0] = ones_complement[0].zfill(len(result[0][0]))
        steps = steps + "<br>Then add 1 to it:<br>" + ones_complement[0] +
"<br>" + add_one + "<br>"+ result[0][0] + "<br>"
        steps = steps + "<br>*Recap of the rules for adding binary numbers,
all you have to remember is that:<br><br>"\
        answer = result[0][0]
    else:
        return ["Error! 1 argument required, given 0 or more than 1", ""]
    return [answer, steps]
```

Function 07: One’s Complement of an Unsigned Integer

```
def one_complement(x, y = None):
    global required_bits
    if len(x) == 1:
        a = x[0]
        a = a.zfill(required_bits)
        temp = ""
        result = "Invert every bit of the given bit string i.e change 0 to 1 and 1
to 0 : <br>" + a + " ==> " +  temp
        return [temp,result]
    elif len(y) == 1:
        temp = bin(int(y[0])).replace("0b", "")
        temp = temp.zfill(required_bits)
        temp2 = ""

        result = decimal_to_binary(y)
        steps = "First, convert decimal number into binary number.<br><br>"

        steps = steps + result[1]
        steps = steps + "<br><br> Then, invert every bit of the given bit string
i.e change 0 to 1 and 1 to 0.<br>" + temp + " ==> " + temp2
        steps = steps + "<br> <br>Finally, convert the result back to decimal
format."

        steps = steps + binary_to_decimal([temp2])[1]
        return ["Base 2: ( " + temp2 + " ) , Base 10: ( " + str(int("0b" + temp2,
base = 2)) + ')', steps]

    else:
        return ["Error! 1 argumnent required, given 0.", ""]
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Function 08: Two's Complement of an Unsigned Integer

```
def twos_complement(binary_numbers, decimal_numbers = None):
    if len(binary_numbers) == 1:
        complement = one_complement(binary_numbers)
        steps = steps + binary_numbers[0] + " ==> " + complement[0]
        bitlen = complement[0].__len__()
        answer = str(bin(int(complement[0], base = 2) + 1).replace('0b', ''))
        answer = answer.zfill(bitlen)
        answer = answer.zfill(required_bits)
        add_one = "1"
        add_one = add_one.zfill(len(answer))
        complement[0] = complement[0].zfill(len(answer))
        steps = steps + "<br><br>Then add 1 to it:<br>" + complement[0] + "
<br>" + add_one + "<br>" + answer
        steps = steps + "<br><br>Recap of the rules for adding binary
numbers, all you have to remember is that:<br><br>"\
        return [answer , steps]

    elif len(decimal_numbers) == 1:
        temp = [bin(decimal_numbers[0]).replace("0b", "")]
        temp[0] = '0' + temp[0]
        complement = one_complement(temp)
        sum = int("0b" + complement[0], base = 2) + 1

        result = decimal_to_binary([decimal_numbers[0]])
        result1 = decimal_to_binary([sum])
        steps = "First, convert decimal number into binary number.<br>"
        steps = steps + result[1]
        steps = steps + " ~ " + result[0] + "<br>(The extra 0 (zero) is always
added at the MSB (Most Significant Bit) of the string whenever we are
dealing with the two's complement. So we have stuffed an extra 0 (zero) at
the MSB of the string.)<br><br>"
        ones_complement = one_complement([result[0]])
        steps = steps + result[0] + " ==> " + ones_complement[0]
        add_one="1"
        add_one = add_one.zfill(len(result[0]))
        steps = steps + "<br><br>Then add 1 to it:<br>" + result[0] + "<br>" +
add_one + "<br>"+ result1[0] + "<br>"
        steps = steps + "<br>*Recap of the rules for adding binary numbers,
all you have to remember is that:<br><br>"\
            "Finally, convert the result back to decimal format. Starting from
LSB, take sum of (2*i)^x where i = bit value and x = bit position.<br><br>
For this,"\
            " follow the following steps: <br>"
        ans = result1[0]
        next_step = "= "
        answer = 0
        if(ans[0]=='1'):
            answer = int(math.pow(2, len(ans)-1)) * -1
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
            next_step = next_step + "- " + str(int(math.pow(2, len(ans)-1)))
        else:
    if(all_numbers[1][1]=="int"):
        if(all_numbers[1][2] == "-"):
            steps = steps + "Operand 2 : Convert to binary: -" +
str(all_numbers[1][0])
            steps = steps + "<br><br>To convert a negative number into
binary: The first step is to convert the positive version of the given
negative number into binary. To perform this, divide the number by 2
repeatedly until remainder becomes smaller than 2. Then read all the carry
in backward (from bottom to top) direction.<br>-" + str(all_numbers[1][0]) +
" ==> " + str(binaries[1]) + "<br><br>"
            steps = steps + str(binaries[1]).zfill(len(result[1][0])) + " ==> " +
result[1][0] + "<br>"
        else:
            steps = steps + "Operand 2 : Convert to binary: " +
str(all_numbers[1][0]) + " ==> " + str(binaries[1]) + "<br>"
        if(all_numbers[1][2] =="-"):
            steps = steps + "<br>Recap of the rules for taking two’s
complement, all you have to remember is that:<br>"\
                    "*Add the extra 0 (zero) at the MSB (Most Significant Bit) of
the string because we are dealing with the two's complement."\

    else:
        if(all_numbers[1][2] == "-"):
            both_neg = both_neg + 1
            steps = steps + "Operand 2 : Base 2( -" + str(all_numbers[1][0]) + "
)<br>"
            #if (two_comp == 1):
            #   steps = steps + "<br>*For a negative number two's complement
of the positive version of the negative number was taken. (Two's
complement is one's complement plus one.)<br><br>"
        else:
            steps = steps + "Operand 2 : Base 2( " + str(all_numbers[1][0]) +
" )<br>"
            #if (two_comp == 2):
    steps = steps + "Adding numbers: <br>"
    explain = ""
    if(isSignednumber):
        steps = steps + "0" + str(op0) + "<br>"
        steps = steps +  "0" + str(op1) + "<br>"
        steps = steps +  str(ans).zfill(len(op0)+1)  + "<br>"
        explain = "<br>*Recap of the rules for adding binary numbers, all you
have to remember is that:<br>"\
        ans = str(ans).zfill(len(op0)+1)
        temp_ans = ans[1:]
        carry = ans[0]
        if (two_comp == 1):
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            steps = steps + "<br>As there was only one negative operand, we
are discarding the carry.<br>" + ans + " ~ " + temp_ans.zfill(len(op0)+1) +
"<br>"
            ans = temp_ans.zfill(len(op0)+1)
        steps = steps + explain
        answer = 0
        steps = steps + explanation
        prev_steps = steps
        steps = "<br>Lastly convert the answer to its decimal value using,
take sum of (2*i)^x where i = bit value and x = bit position: <br><br>For
this, follow the following steps:<br>"
        next_step = "= "
        if(ans[0]=='1'):
            if (two_comp == 1):
                if(carry == '0'):
                    steps = steps +" "+ ans + "=  ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
                else:
                    steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
            else:
                steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
        else:
            steps = steps + ans + " = "+"- ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
            next_step = next_step + "- 0"
        index= len(ans) - 1
        index1= 1
        while (index > 1):
            steps = steps + "+ ( 2^" + str(index-1) + " * " + ans[index1]+" ) "
            if(ans[index1]=="1"):
            else:
                next_step = next_step + " + 0"
            index1 = index1 + 1
            index = index - 1
        steps = steps + "+ ( 2^0"  + " * " + ans[len(ans)-1]+" )<br>"
        if(ans[len(ans)-1] == "1"):
            answer = answer + 1
            next_step = next_step + " + 1<br>"
        else:
            next_step = next_step + " + 0<br>"
        next_step = next_step + " = " + str(answer) + "<br>"
        steps = steps + next_step

    else:
        op0len = len(op0)
        op1len = len(op1)
        anslen = len(ans)
        maxlen = op0len
        if(maxlen<op1len):
            maxlen = op0len
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
    elif(maxlen<anslen):
        maxlen = anslen
    op0=op0.zfill(maxlen)
    op1=op1.zfill(maxlen)
    ans=ans.zfill(maxlen)
    steps = steps +  str(op0) + "<br>"
    steps = steps +  str(op1) + "<br>"
    steps = steps +  str(ans) + "<br>"
        "<br>*Recap of the rules for adding binary numbers, all you have
to remember is that:<br><br>"\
    answer = 0
    steps = steps + explanation
    prev_steps = steps
    steps = "<br>Lastly convert the answer to its decimal value using,
take sum of (2*i)^x where i = bit value and x = bit position: <br><br>For
this, follow the following steps:<br>"
    next_step = "= "
    index= len(ans)
    index1= 0
    steps = steps + ans + " ="
    while (index > 1):
        steps = steps +  " ( 2^" + str(index-1)  + " * " + ans[index1]+" ) + "
        if(ans[index1]=="1"):
        else:
            next_step = next_step + " 0 + "
        index1 = index1 + 1
        index = index - 1
    steps = steps + "( 2^0"  + " * " + ans[len(ans)-1]+" ) <br>"
    if(ans[len(ans)-1] == "1"):
        answer = answer + 1
        next_step = next_step + " 1<br>"
    else:
        next_step = next_step + " 0<br>"
    next_step = next_step + " = " + str(answer) + "<br>"
    steps = steps + next_step
        steps = steps + "- ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
        next_step = next_step + "- 0"
    index= len(ans) - 1
    index1= 1
    while (index > 1):
        steps = steps + "+ ( 2^" + str(index-1) + " * " + ans[index1]+" ) "
        if(ans[index1]=="1"):
        else:
            next_step = next_step + " + 0"
        index1 = index1 + 1
        index = index - 1
    steps = steps + "+ ( 2^0"  + " * " + ans[len(ans)-1]+" )<br>"
    if(ans[len(ans)-1] == "1"):
        answer = answer + 1
        next_step = next_step + " + 1<br>"
    else:
        next_step = next_step + " + 0<br>"
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        next_step = next_step + " = " + str(answer) + "<br>"
        steps = steps + next_step
        final_ans = "Base 2 ( " + ans + " ), Base 10 ( " + str(answer) + ")"

        #steps = steps + binary_to_decimal([result[0]])[1]
        #result[0] = result[0].zfill(required_bits)
        #dec_answer = "Base 2 ( " + result[0] + " ) , Base 10 ( " +str(sum) +
' )'
        return [final_ans, steps]

    else:
        return ["Error! 1 argumnent required, given 0.", ""]
```

Function 09: Bit Representation

```
def bit_representation(decimal_numbers,all_numbers):
    if len(decimal_numbers) == 1:
        steps = "Take log base 2 of the given binary string i.e
log<sub>2</sub>(" + str(decimal_numbers[0]) + " + 1) = " +
str(math.log(decimal_numbers[0]+1, 2))
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        steps = steps + "<br>take ceiling of the previous result like this: 
&lceil;" + str(math.log(decimal_numbers[0] + 1, 2) ) + "&rceil; = " + 
str(math.ceil(math.log(decimal_numbers[0] + 1, 2 )))
        answer = str(math.ceil(math.log(decimal_numbers[0] + 1, 2)))
        if(all_numbers[0][2] == "-"):
            steps = steps + "<br>Since its a negative number: there is atleast 
1 extra bit needed to represent it. So answer: " + 
str(math.ceil(math.log(decimal_numbers[0] + 1, 2)) + 1)
            answer = str(int(answer) + 1)
        steps = steps + "<br> <b>Note: The answer has to be a integer, so 
we round up to the nearest biggest integer.</b>"
        return [answer, steps]

    else:
        return ['Error! 1 arg required, given 0', ""]
```

Function 10: Two’s Complement of a Binary Integer

```
def getBinTwosComplement(num, reqBit):
    steps = "apply one's complement to binary string first and then add 1 to 
LSB (Least Significant Bit)<br>"
    num = num.zfill(reqBit)
    if(userbits > num.__len__()+1):
        num = num.zfill(userbits)
    else:
        num = num.zfill(num.__len__()+1)
    a = num
    temp = ""
    bitlen = temp.__len__()
    answer = str(bin(int(temp, base = 2) + 1).replace('0b', ''))
    if(len(answer) > reqBit):
        answer = answer[1:]
    return [answer , steps ,"bin"]
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Function 11: Two’s Complement of a Decimal Number

```
def getIntTwosComplement(num, reqBit):
    global userbits
    temp = bin(num).replace("0b", "")
    temp=temp.zfill(reqBit)
    a = temp
    temp = ""

    sum = int("0b" + temp, base = 2) + 1

    result = decimal_to_binary([sum])
    binary_res = result[0]

    if(userbits > binary_res.__len__()+1):
        binary_res = binary_res.zfill(userbits)
    steps = steps + result[1]
    steps = steps + "<br> Then, apply one's complement to binary string and then add 1 to LSB (Least Significant Bit) to get its negative representation i.e. " + binary_res
    steps = steps + binary_to_decimal([binary_res])[1]
    return [binary_res, steps,"int", str(sum)]
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Function 12: Get Decimal String

```
def getDecimal(num,isSigned):
    numLen = num.__len__()
    decimal_rep=""
    if(isSigned):
        if(num[0] == "0"):
            decimal_rep = str(int(num, base = 2))
        else:
            num1 = ""
            num1 = num1.zfill(numLen)
            num1 = '1' + num1[1:]
            dec1 = int(num1, base = 2) * -1
            num = '0' + num[1:]
            dec = int(num, base = 2)
            decimal_rep = str(dec1 + dec)
    else:
        decimal_rep = str(int(num, base = 2))
else:
            num1 = ""
            num1 = num1.zfill(numLen)
            num1 = '1' + num1[1:]
            dec1 = int(num1, base = 2) * -1
            num = '0' + num[1:]
            dec = int(num, base = 2)
            decimal_rep = str(dec1 + dec)

    return decimal_rep
```

Function 13: Addition/Subtraction

```
def
binary_add_sub(op_type,all_numbers,binary_numbers,decimal_numbers
= None):
    if(all_numbers.__len__() < 2):
        return ["Error: Not enough operands" ,""]
    isDoubleNeg = False
    global userbits
    global isSignednumber
    two_comp = 0
    isSignednumber = False
    result = []
    explanation= "<br>"
    if(op_type == "subtraction"):
        if(all_numbers.__len__() > 1):
            if(all_numbers[1][2]=="-"):
                isDoubleNeg = True
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            op_type = "addition"
            all_numbers[1][2] = "+"
        else:
            op_type = "addition"
            all_numbers[1][2] = "-"
if(all_numbers[1][2]=="-" or all_numbers[0][2]=="-"):
    isSignednumber = True

binaries=[]
for i in all_numbers:
    if(i[2] == "-"):
        if(i[1] == "int"):
            bin_rep = decimal_to_binary([i[0]])
            r_bit=bin_rep[0].__len__()+1
            if(userbits > r_bit):
                r_bit = userbits
            i.append(r_bit)
            binaries.append(bin_rep[0])
        else:
            r_bit=i[0].__len__()+1
            if(userbits > r_bit):
                r_bit = userbits
            i.append(r_bit)
            binaries.append(i[0])
    else:
        if(i[1] == "int"):
            bin_rep = decimal_to_binary([i[0]])
            r_bit=bin_rep[0].__len__()
            if(userbits > r_bit):
                r_bit = userbits
            i.append(r_bit)
            binaries.append(bin_rep[0])
        else:
            r_bit=i[0].__len__()
            if(userbits > r_bit):
                r_bit = userbits
            i.append(r_bit)
            binaries.append(i[0])
if(all_numbers[0][3] < all_numbers[1][3]):
    if(i[2] == "-"):
        two_comp = two_comp + 1
        if(i[1] == "int"):
            result.append(getIntTwosComplement(i[0],i[3]))
        else:
            result.append(getBinTwosComplement(i[0],i[3]))
    else:
        if(i[1] == "int"):
            bin_rep = decimal_to_binary([i[0]])
            bin_rep[0] = bin_rep[0].zfill(i[3])
            result.append([i[0] , "" ,"bin"])

op0 = result[0][0]
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
    op1 = result[1][0]
    op0len = op0.__len__()
    op1len = op1.__len__()
    if(op0len > op1len):
        op1 = op1.zfill(op0len)
    else:
        op0 = op0.zfill(op1len)
    ans = bin(int(op0, 2) + int(op1, 2)).replace("0b", ")
    op0len = op0.__len__()
    op1len = op1.__len__()
    anslen = ans.__len__()
    ans = ans.zfill(op1len)
    isCarry = False
    isDecimal = False
    dec_value = ""
    if(isSignednumber):
        if(anslen > op1len):
            isCarry = True
    final_ans= ""
    if(all_numbers[0][1] =="int" or all_numbers[1][1]=="int"):
        isDecimal = True
    steps=""
    if(all_numbers[0][1]=="int"):
        if(all_numbers[0][2] == "-"):
            steps = steps + "Operand 1 : Convert to binary: -" +
str(all_numbers[0][0])
            steps = steps + "<br><br>To convert a negative number into
binary: The first step is to convert the positive version of the given
negative number into binary. To perform this, divide the number by 2
repeatedly until remainder becomes smaller than 2. Then read all the carry
in backward (from bottom to top) direction.<br>" + str(all_numbers[0][0]) +
" ==> " + str(binaries[0]) + "<br><br>"
            steps = steps + str(binaries[0]).zfill(len(result[0][0])) + " ==> " +
result[0][0] + "<br>"

        else:
            steps = steps + "Operand 1 : Convert to binary: " +
str(all_numbers[0][0]) + " ==> " + str(binaries[0]) + "<br>"
        if(all_numbers[0][2] =="-"):
            steps = steps + "<br>Recap of the rules for taking two’s
complement, all you have to remember is that:<br>"\
                    "*Add the extra 0 (zero) at the MSB (Most Significant Bit) of
the string because we are dealing with the two's complement."\
    else:
        both_neg = 0
        if(all_numbers[0][2] == "-"):
            steps = steps + "Operand 1 : Base 2( -" + str(all_numbers[0][0]) + "
)<br>"
            both_neg = 1
        else:
            steps = steps + "Operand 1 : Base 2( " + str(all_numbers[0][0]) +
" )<br>"
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
    if(isDoubleNeg):
        steps = steps + "Double negation results in addition i.e. - (-" +
str(all_numbers[1][0]) + ") ==> + " + str(all_numbers[1][0]) + "<br>"
        if(all_numbers[1][2] == "-"):
            steps = steps + "so it is: " + str(all_numbers[0][2]) +
str(all_numbers[0][0]) + " + " + str(all_numbers[1][0]) + "<br><br>"
        else:
            steps = steps + "so it is: " + str(all_numbers[0][0]) + " + " +
str(all_numbers[1][0]) + "<br><br>"
    if(all_numbers[1][1]=="int"):
        if(all_numbers[1][2] == "-"):
            steps = steps + "Operand 2 : Convert to binary: -" +
str(all_numbers[1][0])
            steps = steps + "<br><br>To convert a negative number into
binary: The first step is to convert the positive version of the given
negative number into binary. To perform this, divide the number by 2
repeatedly until remainder becomes smaller than 2. Then read all the carry
in backward (from bottom to top) direction.<br>-" + str(all_numbers[1][0]) +
" ==> " + str(binaries[1]) + "<br><br>"
            steps = steps + str(binaries[1]).zfill(len(result[1][0])) + " ==> " +
result[1][0] + "<br>"
        else:
            steps = steps + "Operand 2 : Convert to binary: " +
str(all_numbers[1][0]) + " ==> " + str(binaries[1]) + "<br>"
        if(all_numbers[1][2] =="-"):
            steps = steps + "<br>Recap of the rules for taking two’s
complement, all you have to remember is that:<br>"\
                    "*Add the extra 0 (zero) at the MSB (Most Significant Bit) of
the string because we are dealing with the two's complement."\

    else:
        if(all_numbers[1][2] == "-"):
            both_neg = both_neg + 1
            steps = steps + "Operand 2 : Base 2( -" + str(all_numbers[1][0]) + "
)<br>"
            #if (two_comp == 1):
            #   steps = steps + "<br>*For a negative number two's complement
of the positive version of the negative number was taken. (Two's
complement is one's complement plus one.)<br><br>"
        else:
            steps = steps + "Operand 2 : Base 2( " + str(all_numbers[1][0]) +
" )<br>"
            #if (two_comp == 2):
    steps = steps + "Adding numbers: <br>"
    explain = ""
    if(isSignednumber):
        steps = steps + "0" + str(op0) + "<br>"
        steps = steps +  "0" + str(op1) + "<br>"
        steps = steps +  str(ans).zfill(len(op0)+1)  + "<br>"
        explain = "<br>*Recap of the rules for adding binary numbers, all you
have to remember is that:<br>"\
        ans = str(ans).zfill(len(op0)+1)
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        temp_ans = ans[1:]
        carry = ans[0]
        if (two_comp == 1):
            steps = steps + "<br>As there was only one negative operand, we are discarding the carry.<br>" + ans + " ~ " + temp_ans.zfill(len(op0)+1) + "<br>"
            ans = temp_ans.zfill(len(op0)+1)
        steps = steps + explain
        answer = 0
        steps = steps + explanation
        prev_steps = steps
        steps = "<br>Lastly convert the answer to its decimal value using, take sum of (2*i)^x where i = bit value and x = bit position: <br><br>For this, follow the following steps:<br>"
        next_step = "= "
        if(ans[0]=='1'):
            if (two_comp == 1):
                if(carry == '0'):
                    steps = steps +" "+ ans + "=  ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
                else:
                    steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
            else:
                steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
        else:
            steps = steps + ans + " = "+"- ( 2^" + str(len(ans)-1) + " * " + ans[0]+" ) "
            next_step = next_step + "- 0"
        index= len(ans) - 1
        index1= 1
        while (index > 1):
            steps = steps + "+ ( 2^" + str(index-1) + " * " + ans[index1]+" ) "
            if(ans[index1]=="1"):
            else:
                next_step = next_step + " + 0"
            index1 = index1 + 1
            index = index - 1
        steps = steps + "+ ( 2^0"  + " * " + ans[len(ans)-1]+" )<br>"
        if(ans[len(ans)-1] == "1"):
            answer = answer + 1
            next_step = next_step + " + 1<br>"
        else:
            next_step = next_step + " + 0<br>"
        next_step = next_step + " = " + str(answer) + "<br>"
        steps = steps + next_step

    else:
        op0len = len(op0)
        op1len = len(op1)
        anslen = len(ans)
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        maxlen = op0len
        if(maxlen<op1len):
            maxlen = op0len
        elif(maxlen<anslen):
            maxlen = anslen
        op0=op0.zfill(maxlen)
        op1=op1.zfill(maxlen)
        ans=ans.zfill(maxlen)
        steps = steps +  str(op0) + "<br>"
        steps = steps +  str(op1) + "<br>"
        steps = steps +  str(ans) + "<br>"
            "<br>*Recap of the rules for adding binary numbers, all you have
to remember is that:<br><br>"\
        answer = 0
        steps = steps + explanation
        prev_steps = steps
        steps = "<br>Lastly convert the answer to its decimal value using,
take sum of (2*i)^x where i = bit value and x = bit position: <br><br>For
this, follow the following steps:<br>"
        next_step = "= "
        index= len(ans)
        index1= 0
        steps = steps + ans + " ="
        while (index > 1):
            steps = steps +  " ( 2^" + str(index-1)  + " * " + ans[index1]+" ) + "
            if(ans[index1]=="1"):
            else:
                next_step = next_step + " 0 + "
            index1 = index1 + 1
            index = index - 1
        steps = steps + "( 2^0"  + " * " + ans[len(ans)-1]+" ) <br>"
        if(ans[len(ans)-1] == "1"):
            answer = answer + 1
            next_step = next_step + " 1<br>"
        else:
            next_step = next_step + " 0<br>"
        next_step = next_step + " = " + str(answer) + "<br>"
        steps = steps + next_step


    if(isDecimal):
        #dec_value = getDecimal(ans,isSignednumber)
        prev_steps = prev_steps + steps
        steps = prev_steps
        final_ans = "Base 2 ( " + ans + "), Base 10 ( " + str(answer) + " )"
    if(all_numbers[1][1]=="int"):
        if(all_numbers[1][2] == "-"):
            steps = steps + "Operand 2 : Convert to binary: -" +
str(all_numbers[1][0])
            steps = steps + "<br><br>To convert a negative number into
binary: The first step is to convert the positive version of the given
negative number into binary. To perform this, divide the number by 2
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
repeatedly until remainder becomes smaller than 2. Then read all the carry
in backward (from bottom to top) direction.<br>-" + str(all_numbers[1][0]) +
" ==> " + str(binaries[1]) + "<br><br>"
            steps = steps + str(binaries[1]).zfill(len(result[1][0])) + " ==> " +
result[1][0] + "<br>"
        else:
            steps = steps + "Operand 2 : Convert to binary: " +
str(all_numbers[1][0]) + " ==> " + str(binaries[1]) + "<br>"
        if(all_numbers[1][2] =="-"):
            steps = steps + "<br>Recap of the rules for taking two’s
complement, all you have to remember is that:<br>"\
                    "*Add the extra 0 (zero) at the MSB (Most Significant Bit) of
the string because we are dealing with the two's complement."\

    else:
        if(all_numbers[1][2] == "-"):
            both_neg = both_neg + 1
            steps = steps + "Operand 2 : Base 2( -" + str(all_numbers[1][0]) + "
)<br>"
            #if (two_comp == 1):
            #   steps = steps + "<br>*For a negative number two's complement
of the positive version of the negative number was taken. (Two's
complement is one's complement plus one.)<br><br>"
        else:
            steps = steps + "Operand 2 : Base 2( " + str(all_numbers[1][0]) +
" )<br>"
            #if (two_comp == 2):
    steps = steps + "Adding numbers: <br>"
    explain = ""
    if(isSignednumber):
        steps = steps + "0" + str(op0) + "<br>"
        steps = steps +  "0" + str(op1) + "<br>"
        steps = steps +  str(ans).zfill(len(op0)+1)  + "<br>"
        explain = "<br>*Recap of the rules for adding binary numbers, all you
have to remember is that:<br>"\
        ans = str(ans).zfill(len(op0)+1)
        temp_ans = ans[1:]
        carry = ans[0]
        if (two_comp == 1):
            steps = steps + "<br>As there was only one negative operand, we
are discarding the carry.<br>" + ans + " ~ " + temp_ans.zfill(len(op0)+1) +
"<br>"
            ans = temp_ans.zfill(len(op0)+1)
        steps = steps + explain
        answer = 0
        steps = steps + explanation
        prev_steps = steps
        steps = "<br>Lastly convert the answer to its decimal value using,
take sum of (2*i)^x where i = bit value and x = bit position: <br><br>For
this, follow the following steps:<br>"
        next_step = "= "
        if(ans[0]=='1'):
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            if (two_comp == 1):
                if(carry == '0'):
                    steps = steps +" "+ ans + "=  ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
                else:
                    steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
            else:
                steps = steps +" "+ ans + "= - ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
        else:
            steps = steps + ans + " = "+"- ( 2^" + str(len(ans)-1) + " * " +
ans[0]+" ) "
            next_step = next_step + "- 0"
        index= len(ans) - 1
        index1= 1
        while (index > 1):
            steps = steps + "+ ( 2^" + str(index-1) + " * " + ans[index1]+" ) "
            if(ans[index1]=="1"):
            else:
                next_step = next_step + " + 0"
            index1 = index1 + 1
            index = index - 1
        steps = steps + "+ ( 2^0"  + " * " + ans[len(ans)-1]+" )<br>"
        if(ans[len(ans)-1] == "1"):
            answer = answer + 1
            next_step = next_step + " + 1<br>"
        else:
            next_step = next_step + " + 0<br>"
        next_step = next_step + " = " + str(answer) + "<br>"
        steps = steps + next_step

    else:
        op0len = len(op0)
        op1len = len(op1)
        anslen = len(ans)
        maxlen = op0len
        if(maxlen<op1len):
            maxlen = op0len
        elif(maxlen<anslen):
            maxlen = anslen
        op0=op0.zfill(maxlen)
        op1=op1.zfill(maxlen)
        ans=ans.zfill(maxlen)
        steps = steps +  str(op0) + "<br>"
        steps = steps +  str(op1) + "<br>"
        steps = steps +  str(ans) + "<br>"
            "<br>*Recap of the rules for adding binary numbers, all you have
to remember is that:<br><br>"\
        answer = 0
        steps = steps + explanation
        prev_steps = steps
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        steps = "<br>Lastly convert the answer to its decimal value using, take sum of (2*i)^x where i = bit value and x = bit position: <br><br>For this, follow the following steps:<br>"
        next_step = "= "
        index= len(ans)
        index1= 0
        steps = steps + ans + " ="
        while (index > 1):
            steps = steps +  " ( 2^" + str(index-1)  + " * " + ans[index1]+" ) + "
            if(ans[index1]=="1"):
            else:
                next_step = next_step + " 0 + "
            index1 = index1 + 1
            index = index - 1
        steps = steps + "( 2^0"  + " * " + ans[len(ans)-1]+" ) <br>"
        if(ans[len(ans)-1] == "1"):
            answer = answer + 1
            next_step = next_step + " 1<br>"
        else:
            next_step = next_step + " 0<br>"
        next_step = next_step + " = " + str(answer) + "<br>"
        steps = steps + next_step
    else:
        steps = prev_steps
        final_ans = ans

    return [final_ans, steps]
```

Function 14: Controller

```
def binary_module(query):
    global required_bits
    global userbits
    global default_bits
    global isSignednumber
    required_bits= 4
    default_bits = 0
    isSignednumber = False
    userbits = 0
    isuserbit = False
    tokenize = nltk.word_tokenize(query)
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
    for word in tokenize:
        bitindex = word.find("bit")
        bitindex1 = word.find("bits")
        if(bitindex > -1 and bitindex1 < 0):
            if(bitindex > 0):
                isuserbit = True
            user_bit= int(word[:bitindex])
            userbits= user_bit
            if(user_bit > required_bits):
                required_bits = user_bit
    if(isuserbit == False):
        required_bits=0
    kwd = [x for x in tokenize if x in keyword_list] # get common keywords

    binar_numbers = []
    binar_signed =[]
    decimal_numbers = []
    decimal_signed =[]
    all_numbers=[]
    # isolate decimal and binary number arguments.
    for i in tokenize:
        x = i
        isSigned = False
        if i[0]== '-':
            isSigned = True
        if re.match(r'[0-9]', i):
            if check_binary_format(i) == "yes" and "base10" not in query and
"base ten" not in query:
                if(i.find("bit") < 0 ):
                    if ((len(i) < required_bits) and ('sum' not in kwd and '+'  not in
kwd and "difference" not in kwd and "-" not in kwd)):
                        i.zfill(required_bits)
                    binar_numbers.append(i)
                    binar_signed.append(isSigned)
                    if(isSigned):
                        all_numbers.append([i,"bin","-"])
                    else:
                        all_numbers.append([i,"bin","+"])

            else:
                if(i.find("bit") < 0 ):
                    decimal_numbers.append(int(i))
                    decimal_signed.append(isSigned)
                    if(isSigned):
                        all_numbers.append([int(i),"int","-"])
                    else:
                        all_numbers.append([int(i),"int","+"])

    # print(binar_numbers, decimal_numbers)

    try:
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        if (('one' in kwd or 'ones' in kwd or "one's" in kwd) or ('two' not in kwd
and 'twos' not in kwd and "two's" not in kwd)) and ('compliment' in kwd or
'complement' in kwd):
            if(all_numbers[0][2] == "-"):
                return signed_ones_complement(decimal_numbers,
all_numbers)
            else:
                return one_complement(binar_numbers, decimal_numbers)

        elif ('two' in kwd or 'twos' in kwd or "two's" in kwd) and ('compliment'
in kwd or 'complement' in kwd):
            if(all_numbers[0][2] == "-"):
                return signed_twos_complement(decimal_numbers,
all_numbers)
            else:
                return twos_complement(binar_numbers, decimal_numbers)

        elif 'bits' in kwd:
            return bit_representation(decimal_numbers,all_numbers)

        elif 'sum' in kwd or '+' in kwd:
            return
binary_add_sub("addition",all_numbers,binar_numbers,decimal_numbers)
        elif 'difference' in kwd or '-' in kwd:
            return
binary_add_sub("subtraction",all_numbers,binar_numbers,decimal_numb
ers)

        elif 'convert' in kwd or 'write' in kwd or 'represent':
            if "to decimal" in query or "binary to decimal" in query or 'decimal'
in query:
                return binary_to_decimal(binar_numbers, decimal_numbers)
            elif "to binary" in query or "decimal to binary" in query or 'binary' in
query:
                if(all_numbers[0][2] == "-"):
                    return
signed_decimal_to_binary(decimal_numbers,all_numbers)
                else:
                    return decimal_to_binary(decimal_numbers)
        else:
            return ["query format not correct, please repeat the question
again.", ""]

    except:
        # raise Exception
        return ["Sorry, can you repeat your question", ""]
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

APPENDIX D. SERVER

```
    if len(compare(kwd)) > 0:
        result = binary_module(kwd)
        result1 = ""
        response = ""
        if(len(result) > 1):
            result1 = result[1]
            response = '{"result" : "' + result[0] + '", "steps": "<br
/><b>STEPS:</b> <br/>' + result1 + '"}'
        else:
            response = '{"result" : "' + result[0] + '"}'
        resp = make_response(response)
        print(addDataToMongo(kwd, result[0], db))
    elif kwd == 'help':
        guide = "<span class= 'text-center'><b>User Help
Guide</span></b><br><br>Hi! My name is <i>Serina</i>, and welcome to
the Help Guide.It can also perform signed binary calculation where we are
using the two’s complement of the number to represent its negative
number. User can also specify the number of bits in which the answer is
required. If the user specified bits are less than minimum bits required for
answer, the answer will be shown in minimum required bits.<br><br>The
following are the sample questions. You can follow any of these formats to
get the results:<br><br><b>How many bits</b><br>how many bits are
required to represent 37 in binary?<br>how many bits are required to
represent -37 in binary?"\
```

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
        "<br>how many bits are required to represent 111 in binary in base10?<br>how many bits are required to represent -111?<br>How many bits in -16?<br>How many bits in 32?
        "<br><br><b>One's complement of binary and decimal numbers</b><br>what's the one's complement of 1010?<br>what's the one's complement of 1010 in base10?  <br>what's the one's complement of 12?<br>what's the one's complement of -12? "\
        "<br><br><b>Two's complement of binary and decimal numbers</b><br>what's the two's complement of -1010?<br>what's the two's complement of -1010 in base ten?  <br>what's the two's complement of 24?<br>what's the two's complement of -24?"\
        "<br><br><b>Addition and subtraction questions in two's complement</b> <br>What will be the answer for -1001 - 01011?<br>What will be the answer for -1010 + 0110?<br>What will be the answer for 1001 + -0110?<br>What will be the answer for 1001 - -0110?<br>what's the sum of 10101 and 11?<br>what will be the answer for 11010 - 001?<br>what will be the answer for -2 + 5?<br>what will be the answer for 2 - 5?<br>what will be the answer for 2 - -5?<br>what will be the answer for -2 - -5?"\
        "<br><br><b>Shortened forms of the questions will also be functional and provide correct results</b><br>1001 - 01011<br>-1010 + 0110<br>1001 + -0110<br>1001 -0110<br>-29
        <b>Also you can speccify how many bits (Note: There is no maximum and minimum. It depends on the question. If a question should be done in 5 bits you can specify any number of bits. If the specified bits are less than 5 it will give the answer in the most accurate bits i.e. 5.  If the specified bits are greater than 5 lets say 10, then it will give answer in 10 bits.)</b><br>what's the one's complement of 12 in 13bit?"\
        "<br>what's the one's complement of -14 in 13bit?<br>what's the two's complement of 9 in 16bit?<br>what will be the answer for -2 + 5 in 10bit?<br>-2 + 5 in 10bit? <br>7- -3  in 8bit "\
        "<br><br><b>Spacing requirements needed for addition and subraction</b><br><br>Spaces are required between operator and operands. If the space is not provided for example 100-10 or 100+10, it will be considered as a magnitude of the respective operand. Thus a error message will appear."\
        "<br><br>By Default, it takes numbers as binary if it only consists on 1’s and 0’s, e.g. 1, 11, 101.If you want them to be considered decimal, add keyword ‘base10’ or ‘base ten’ along, e.g. '24 - 10 in base10'<br><br><b>Common error messages sources and their solutions</b>"\
        "<br><br>Below are questions that will result in a error messsage: Sorry,  can you please repeat that? You can type 'help' for assistance. With the reasons for these errors, and solutions to them <br><br><b>1. Typing: convert 101 to binary?</b>"\
        "<br><br>Will give you an error message because 101 is a binary number by default and that question calculates the bits required to store a decimal number in binary."\
        "<br><br><b>Solution:</b> <br>In order to get the correct answer you must type <br>Convert 101 base10 to binary? Or Convert 101 base ten to binary?<br><br><b>2. Typing: how many bits in 100?</b>"\
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
            "<br><br>Will give you an error message because 100 is a binary
number by default and that question calculates the bits required to store a
decimal number in binary. <br><br><b>Solution:</b><br>In order to get
the correct answer you must type <br>How many bits in 100 base10? Or
How many bits in 100 base ten?"\
            "<br><br><b>3. Typing 1’s, 1s, 2’s and 2s as keywords for one’s
and two’s complement. Will result in a error message, because the two
sets of numbers in the question will confuse the chatbot as what operation
will need to be done.</b><br><br><b>Soltuion: </b><br>Use the text
(one's, ones or one) for one's complement and  (two's, twos or two) for
two's complement."
        result = '{"result" : "' + guide + '"}'
        resp = make_response(result)
    else:
        lex = boto3.client(
            'lex-runtime',
        )
        response = lex.post_text (
            botName='UniChatBot',
            botAlias='aliasTwo',
            userId='655701873205',
            sessionAttributes={
                'string': 'string'
            },
            requestAttributes={
                'string': 'string'
            },
            inputText= request.form['keyword']

        lexResponse = response['message'].replace('"', ""
        print('response', type(response), response.__str__())
        #if '|' in lexResponse or 'noun' in lexResponse or 'verb' in
lexResponse:
        #    result = '{"result" : ' + lexResponse + '}'
        #else:
        lexResponse = lexResponse.replace("\\n", "<br/>")
        result = '{"result" : "' + lexResponse + '"}'
        print ("result " + result)
        resp = make_response(result)
        print(addDataToMongo(kwd, lexResponse, db))
        print ('resp', resp)
    resp.status_code = 200
```

APPENDIX E. LEX

```
#------------------------ for lex bot 'TestBot' ----------------------#
# lex = boto3.client('lex-runtime')
# response = lex.post_text(
#     botName='TestBot',
```

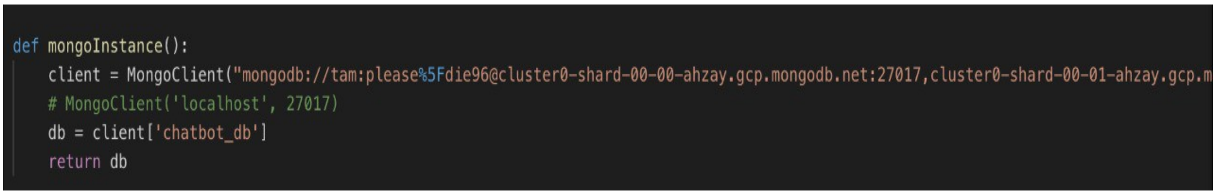
This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

```
#     botAlias='aliasOne',
#     userId='655701873205',
#     sessionAttributes={
#         'string': 'string'
#     },
#     requestAttributes={
#         'string': 'string'
#     },
#     inputText='hi'
# )
# print(response['message'])
# -------------------- for lex bot 'UniChatBot' ----------------------#
lex = boto3.client(
    'lex-runtime',
)
response = lex.post_text(
    botName='UniChatBot',
    botAlias='aliasTwo',
    userId='655701873205',
    sessionAttributes={
        'string': 'string'
    },
    requestAttributes={
        'string': 'string'
    },
    inputText='can you tell'
)
print(response['message'])
```

APPENDIX F. Atlas Database

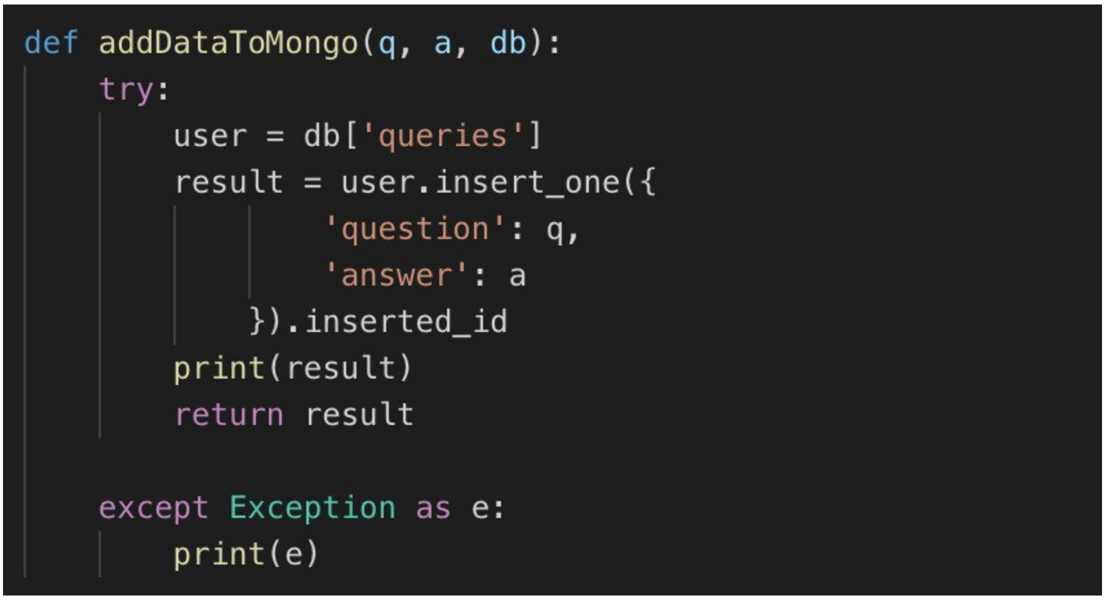
Create Mongo dB Instances



```
def mongoInstance():
    client = MongoClient("mongodb://tam:please%5Fdie96@cluster0-shard-00-00-ahzay.gcp.mongodb.net:27017,cluster0-shard-00-01-ahzay.gcp.m
    # MongoClient('localhost', 27017)
    db = client['chatbot_db']
    return db
```

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Adding data to Mongo database



```python
def addDataToMongo(q, a, db):
    try:
        user = db['queries']
        result = user.insert_one({
                'question': q,
                'answer': a
            }).inserted_id
        print(result)
        return result

    except Exception as e:
        print(e)
```

REFERENCES

[1] P. H. S. B. Alice Kerlyl, "Bringing Chatbots into education: Towards Natural Language Negotiation of Open Learner Models," in Applications and Innovations in Intelligent Systems XIV, London, Springer, London, 2007, pp. 179-192.

[2] A. Gelbukh, "Natural language processing," in Fifth International Conference on Hybrid Intelligent Systems (HIS'05), Rio de Janeiro, Brazil, 2005.

[3] V. N. Gudivada, "Natural Language Core Tasks and Applications," in

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)

Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications, Elsevier B.V., 2018, pp. 403-428.

[4] C. B. L. H. N. Lo, "Chatbots and conversational agents: A bibliometric analysis," in 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, 2017.

[5] N. Haristiani, "Artificial Intelligence (AI) Chatbot as Language Learning Medium: An inquiry," in Journal of Physics: Conference Series, Volume 1387, International Conference on Education, Science and Technology, Padang, Indonesia, 2019.

[6] C.-T. L. J.-Y. N. M. Z. D. Z. R. Y. Yiping Song, "An Ensemble of Retrieval-Based and Generation-Based Human-Computer Conversation Systems," in Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18), 2018.

[7] R. Winkler, "Unleashing the Potential of Chatbots in Education: A State-Of-The-Art Analysis," in 78th annual meeting of the academy of management, Chicago, Illinois, 2018.

[8] L. W. &. K. D. R. Anderson, A taxonomy for learning, teaching and assessing: A revision of Bloom's taxonomy of educational objectives, NewYork, 2001.

[9] E. A. Bayan Abu Shawar, "Chatbots: Are they Really Useful?," in LDV Forum 2007, 2007.

[10] A. C. I. &. A. V. Pinar Saygin, "Turing Test: 50 Years Later," Minds and Machines, vol. 10, p. 463–518, 2000.

[11] J. Weizenbaum, "ELIZA—A computer program for the study of natural language communication between man and machine," Communications of the ACM, vol. 9, no. 1, pp. 36-45, 1966.

[12] R. S. Wallace, " The Anatomy of A.L.I.C.E," in Parsing the Turing Test, Dordrecht, Springer, Dordrecht, 2009, pp. 181-210.

[13] G. &. Z. S. Molnár, "The Role of Chatbots in Formal Education," in IEEE 16th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia, 2018.

Downloaded by HASSAN ALI (hxn112233@gmail.com)

[14] A. L. S. B. D. G. E. T. M. David Ferrucci, "Watson: Beyond Jeopardy!," Artificial Intelligence, vol. 199–200, pp. 93-105, 2013.

[15] VENTUREBEAT, "Universities deploy digital assistants to enroll and support students," TECHNOLOGY BREAKING NEWS, 30 January 2019. [Online]. Available: https://www.technologybreakingnews.com/2019/01/universities-deploy-digital-assistants-to-enroll-and-support-students/. [Accessed 12 5 2019].

[16] L. P. a. B. Castleman, "Summer Melt," Harvard Education Press, 2014. [Online]. Available: https://success.gsu.edu/initiatives/reduction-of-summer-melt/. [Accessed 6 7 2019].

[17] S. Z. György Molnár, "The Role of Chatbots in Formal Education," in IEEE 16th International Symposium on Intelligent Systems and Informatics, 2018.

[18] A. Hussain, "5 use cases for a campus chatbot," 19 August 2018. [Online]. Available: http://www.aftabhussain.com/chatbot_uses.html. [Accessed 2 5 2019].

This document is available free of charge on studocu

Downloaded by HASSAN ALI (hxn112233@gmail.com)