REPORT FOR “WHIZ – AI SOLVER”

As a project work for Course

ARTIFICIAL INTELLIGENCE (INT 404)

Name	: Amit Kumar
Registration Number	: 11805621
Name	: Shruti
Registration Number	: 11804020
Program	: CSE B.Tech
Semester	: Fourth
School	: School of Computer Science and Engineering
Name of the University	: Lovely Professional University
Date of submission	:30th March 2020

Lovely Professional University
Jalandhar, Punjab, India.

Transforming Education Transforming India

CRYTPARITHEMATIC SOLVER

ABSTRACT:-

The term CryptArithm: it is a mathematical puzzle which involves the replacement of digits with alphabets, symbols and letters. The art and science of solving CryptArithms is known as Cryptarithmetic. CryptArithmetic is related to constraint satisfaction problems (CSP) which uses Operators (Arithmetic) e.g +, -, between relevant words to make mathematical relations in a way that final outcome is logically true. It also ensures that the assigning of values every letter of each word and numbers will produce the accurate result. The numbers are assigned in such a way that no letter should have two numbers (0...9), each letter should contain different number from 0 to 9. Solving CryptArithmetic generally includes a mixed approach of derivation and huge possibility tests. This paper is an effort to find a solution of a Classical CryptArithmetic Problem (CAT+RAN=AWAY) which is computationally expensive and time consuming by using parallel genetic algorithm.

ACKNOWLEDGEMENT:-

I would like to thank my mentor - Prof. Sagar Pande for his advice and inputs on this project. Many thanks to my friends and seniors as well, who spent countless hours to listen and provide feedbacks.

Table of Contents

INTRODUCTION:-

1.1 Context

This project has been done as part of my course for the CSE at Lovely Professional University . Supervised by Sagar Pande, I have three months to fulfill the requirements in order to succeed the module.

1.2 Idea:-

As a first experience, we wanted to make my project as much didactic as possible by approaching every different steps of the various algorithms and trying to understand them deeply. The goal was to solve all the various constraint satisfaction problems asked in various placement drive examinations as efficiently as possible.

1.3 Problem definition:-

We are working on a crypt-arithmetic solver to add two or more numbers. I've used brute-force to find all the permutations and tkinter package for implementing the gui. Cryptarithmetic problems are mathematical puzzles in which the digits are replaced by letters of the alphabets.

By using standard arithmetic rules we need to decipher the alphabet. Each alphabet takes only one number from 0 to 9 uniquely. After replacing letters by their digits, the resulting arithmetic operations must be correct.

Example :

```
   SEND               9567
 +MORE        =      +1085
-----------       ---------
  MONEY              10652
```

TEAM MEMBERS:-

TEAM LEADER:-

Amit Kumar:-

Contributions:-

1. Algorithm Designing
2. Research
3. GUI
4. Reports

Shruti:-

Contributions:-

1. Ideas
2. Improvements
3. Reviewing
4. Testing Cases

LIBRARIES:-

Itertools:-

Python’s Itertool is a module that provides various functions that work on iterators to produce complex iterators. This module works as a fast, memory-efficient tool that is used either by themselves or in combination to form **iterator algebra**.
For example, let’s suppose there are two lists and you want to multiply their elements. There can be several ways of achieving this. One can be using the naive approach i.e by iterating through the elements of both the list simultaneously and multiply them. And another approach can be using the map function i.e by passing the mul operator as a first parameter to the map function and Lists as the second and third parameter to this function. Let’s see the time taken by each approach.

Itertools.permutation()

Itertools.permutation() function falls under the Combinatoric Generators. The recursive generators that are used to simplify combinatorial constructs such as permutations, combinations, and Cartesian products are called combinatoric iterators.

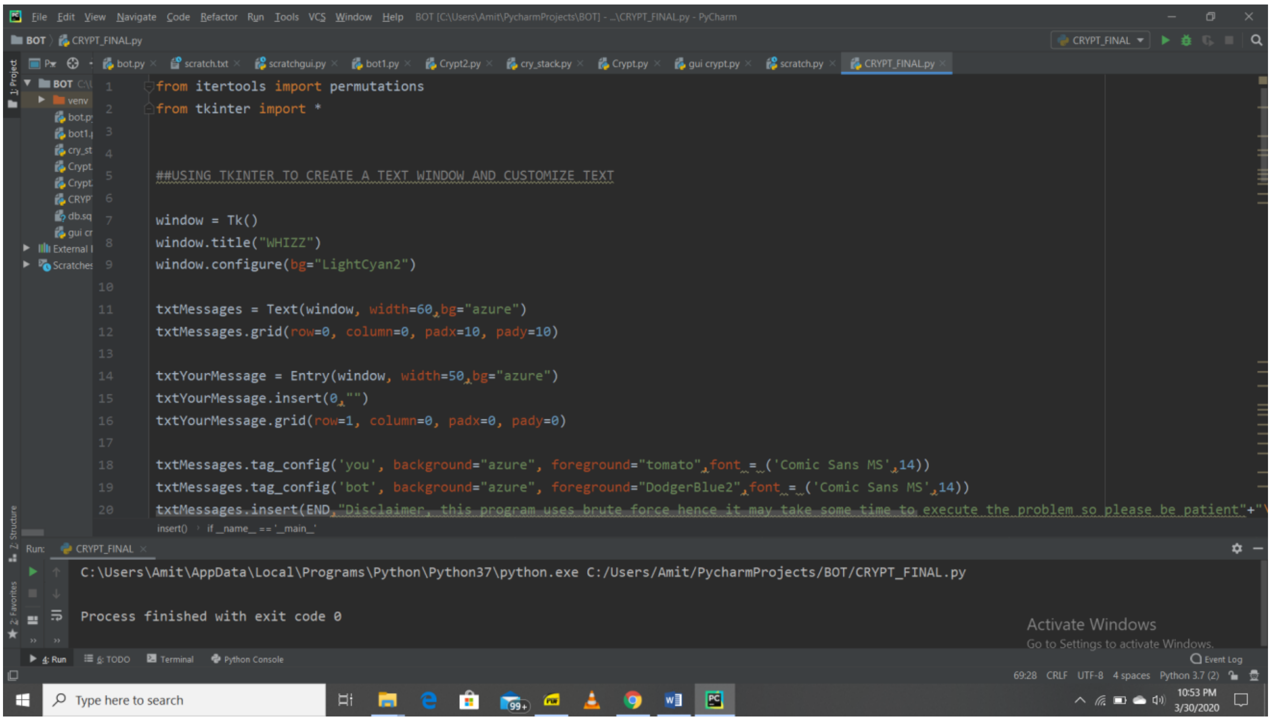
As understood by the word “Permutation” it refers to all the possible combinations in which a set or string can be ordered or arranged. Similarly
here itertool.permutations() method provides us with all the possible arrangements that can be there for an iterator and all elements are assumed to be unique on the basis of there position and not by there value or category. All these permutations are provided in lexicographical order. The function itertool.permutations() takes an iterator and ‘r’ (length of permutation needed) as input and assumes ‘r’ as default length of iterator if not mentioned and returns all possible permutations of length ‘r’ each.

Tkinter:-

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

SCREENSHOTS:-

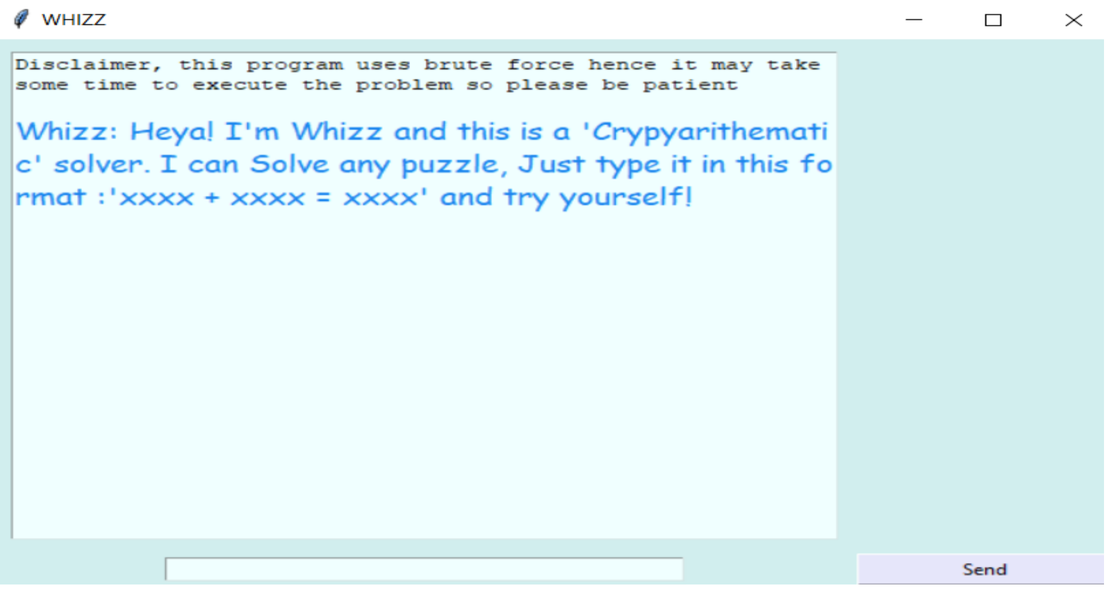
Pycharm Enviroment:


File Edit View Navigate Code Refactor Run Tools VCS Window Help BOT [C:\Users\Amit\PycharmProjects\BOT] - ...\CRYPT_FINAL.py - PyCharm
BOT 〉 CRYPT_FINAL.py
CRYPT_FINAL
bot.py scratch.txt scratchgui.py bot1.py Crypt2.py cry_stack.py Crypt.py gui crypt.py scratch.py CRYPT_FINAL.py
BOT venv bot.py bot1.py cry_st Crypt Crypt CRYPT db.sq gui cr External Scratches
insert() › if __name__ == '__main__'
Run: CRYPT_FINAL
4: Run 6: TODO Terminal Python Console Event Log
69:28 CRLF UTF-8 4 spaces Python 3.7 (2)
Activate Windows
Go to Settings to activate Windows.
Type here to search
10:53 PM
3/30/2020

```python
from itertools import permutations
from tkinter import *


##USING TKINTER TO CREATE A TEXT WINDOW AND CUSTOMIZE TEXT

window = Tk()
window.title("WHIZZ")
window.configure(bg="LightCyan2")

txtMessages = Text(window, width=60,bg="azure")
txtMessages.grid(row=0, column=0, padx=10, pady=10)

txtYourMessage = Entry(window, width=50,bg="azure")
txtYourMessage.insert(0,"")
txtYourMessage.grid(row=1, column=0, padx=0, pady=0)

txtMessages.tag_config('you', background="azure", foreground="tomato",font = ('Comic Sans MS',14))
txtMessages.tag_config('bot', background="azure", foreground="DodgerBlue2",font = ('Comic Sans MS',14))
txtMessages.insert(END,"Disclaimer, this program uses brute force hence it may take some time to execute the problem so please be patient"+"
```

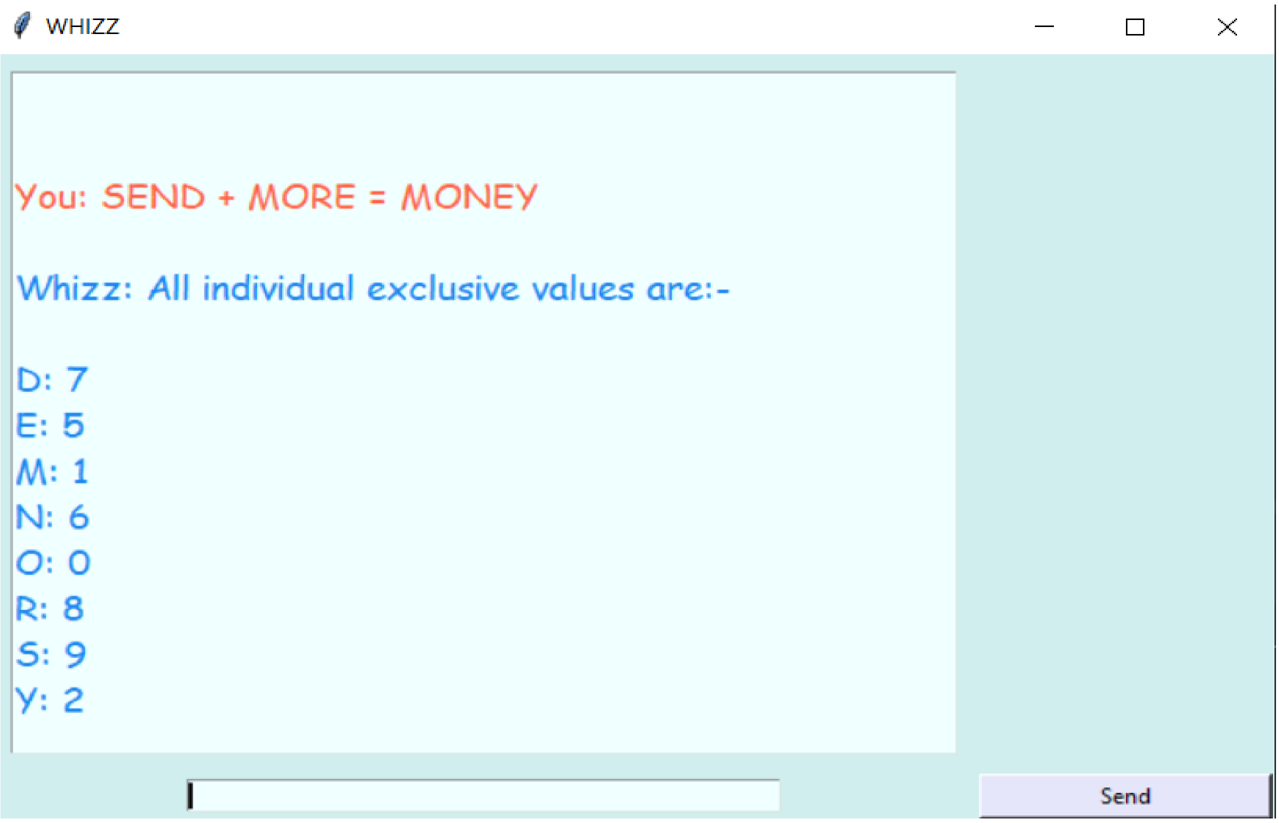
```
C:\Users\Amit\AppData\Local\Programs\Python\Python37\python.exe C:/Users/Amit/PycharmProjects/BOT/CRYPT_FINAL.py

Process finished with exit code 0
```

UI :

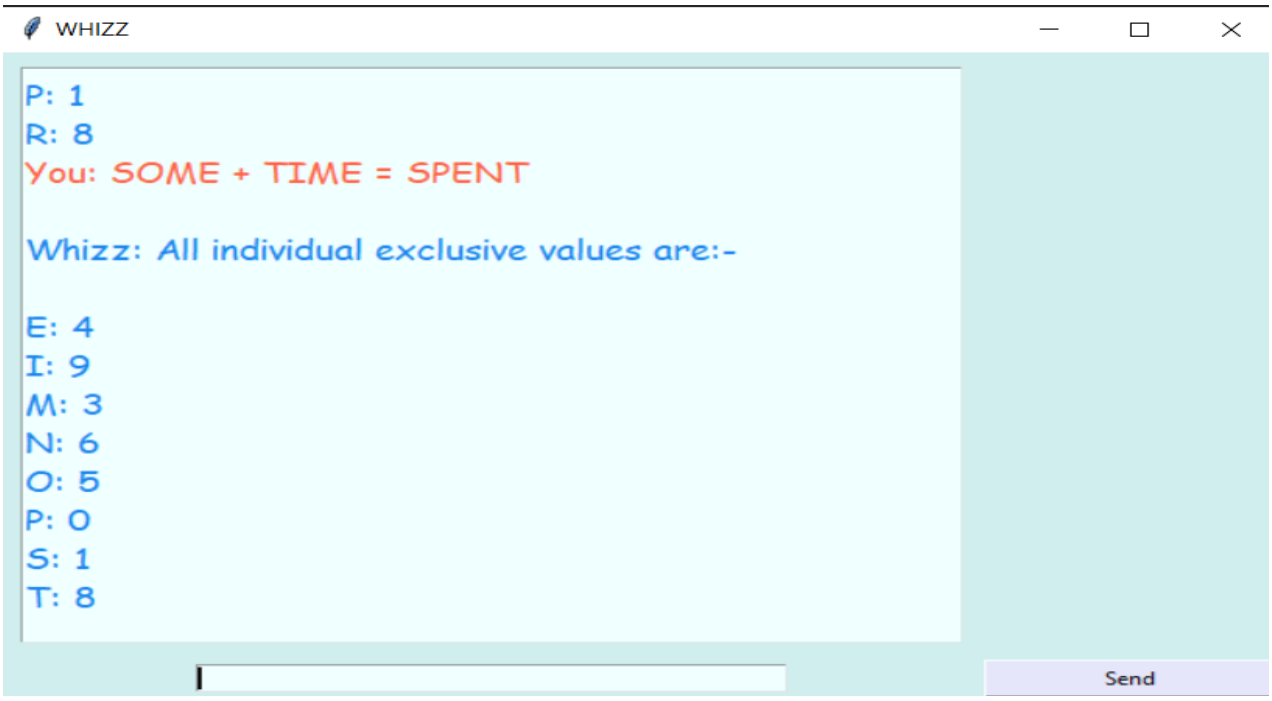


```
Disclaimer, this program uses brute force hence it may take
some time to execute the problem so please be patient

Whizz: Heya! I'm Whizz and this is a 'Crypyarithemati
c' solver. I can Solve any puzzle, Just type it in this fo
rmat :'xxxx + xxxx = xxxx' and try yourself!
```

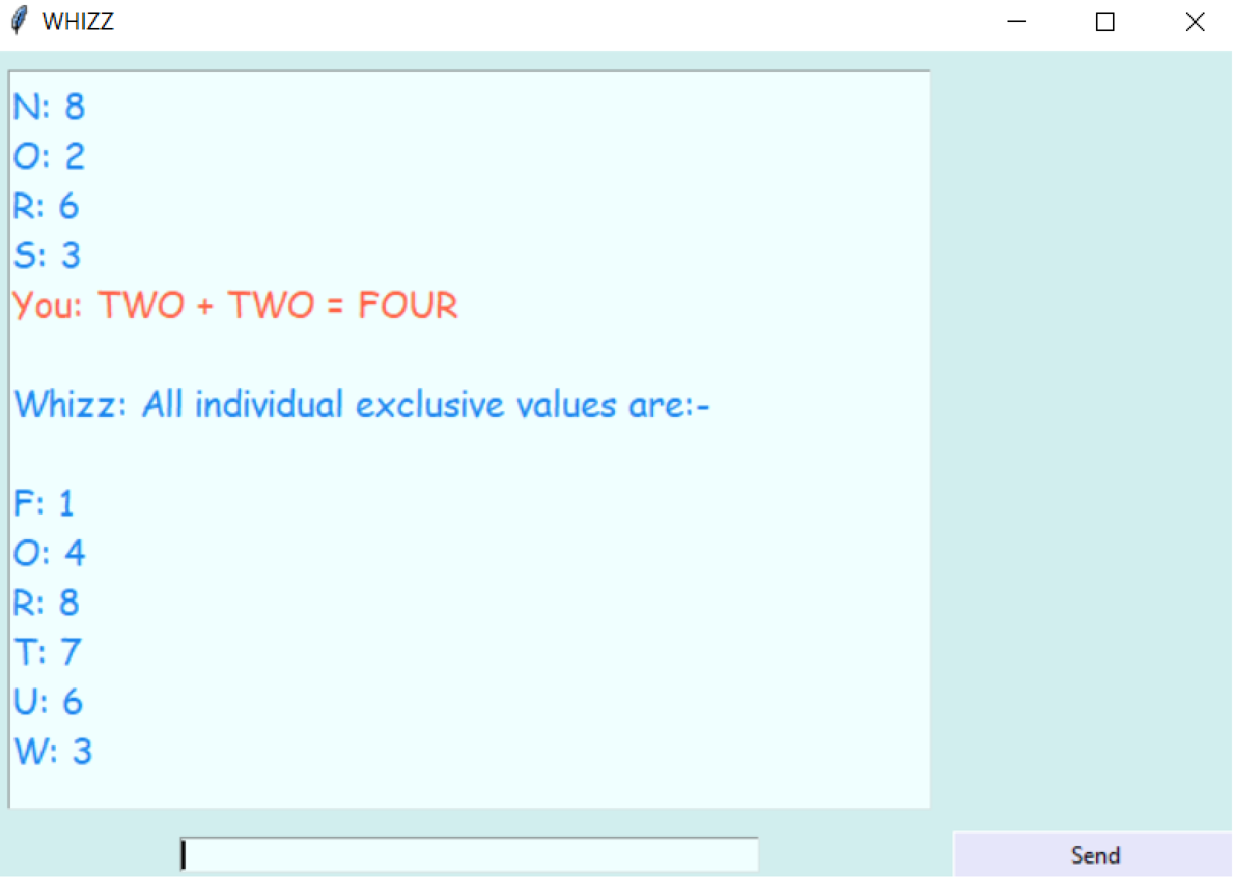
Sample runs :



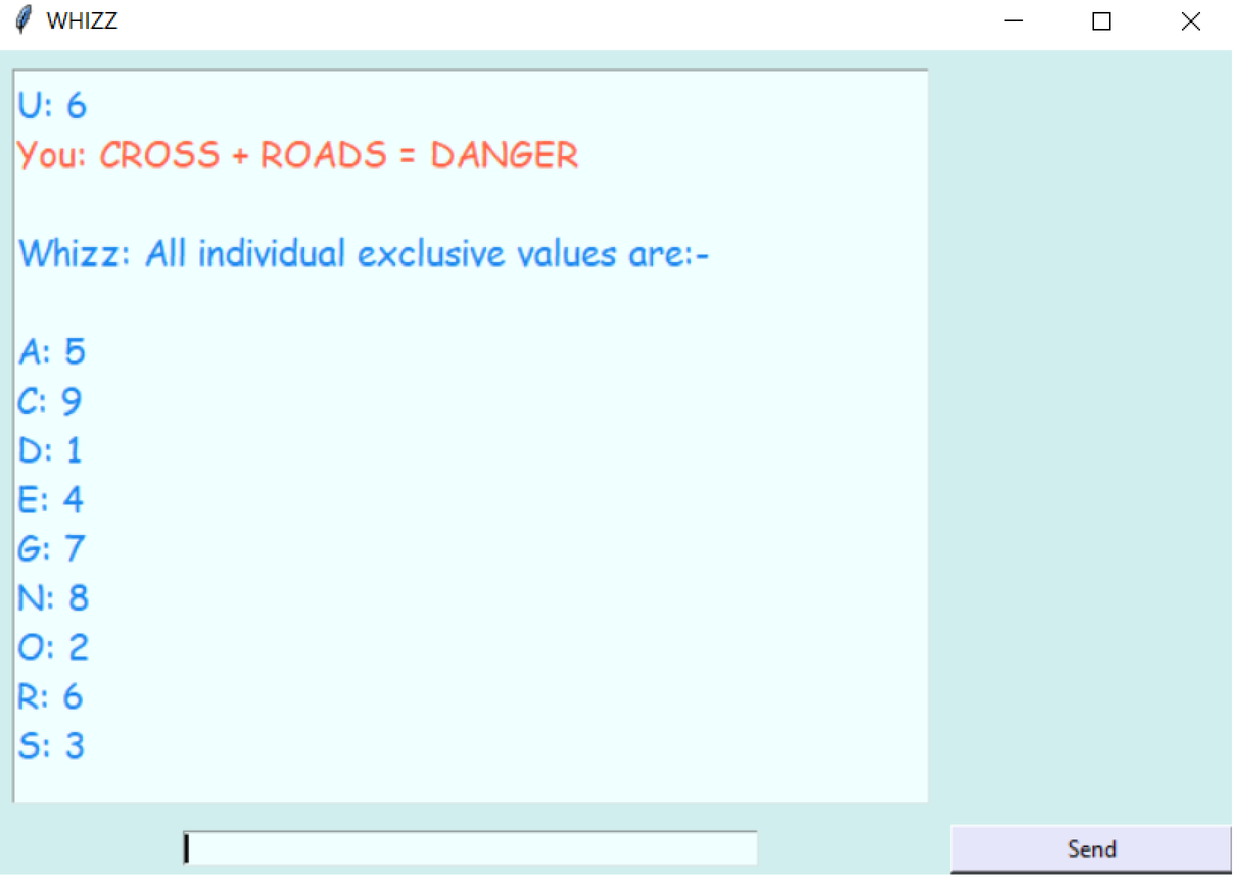
```
You: SEND + MORE = MONEY

Whizz: All individual exclusive values are:-

D: 7
E: 5
M: 1
N: 6
O: 0
R: 8
S: 9
Y: 2
```



```
P: 1
R: 8
You: SOME + TIME = SPENT

Whizz: All individual exclusive values are:-

E: 4
I: 9
M: 3
N: 6
O: 5
P: 0
S: 1
T: 8
```



```
N: 8
O: 2
R: 6
S: 3
You: TWO + TWO = FOUR

Whizz: All individual exclusive values are:-

F: 1
O: 4
R: 8
T: 7
U: 6
W: 3
```



```
U: 6
You: CROSS + ROADS = DANGER

Whizz: All individual exclusive values are:-

A: 5
C: 9
D: 1
E: 4
G: 7
N: 8
O: 2
R: 6
S: 3
```

TEST-CASES:-

A test case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application.

Test cases:

s.no	Test Case	Result
1.	TWO + TWO = FOUR	PASS
2.	CROSS + ROADS = DANGER	PASS
3.	SEND + MORE = MONEY	PASS
4.	LOGIC + LOGIC = PROLOG	PASS
5.	EGG + EGG = PAGE	PASS
6.	BASE + BALL = GAMES	PASS
7.	YOUR + YOU = HEART	PASS
8.	SOME + TIME = SPENT	PASS
9.	EAT + THAT = APPLE	PASS
10.	CP + IS + FUN = TRUE	PASS

Conclusions:-

It is our team's hope that this document will be of huge help with understanding of our little project as we have used an approach which has proved beneficial for us and easy for us to understand the vast ocean that is Artificial Intelligence. We were successful in solving almost all known constraint satisfaction problems known to us. The only downfall of this project was the time complexity as it uses a brute force approach hence to compute all the permutations it takes a huge lot of time, we are working forward to improve it even more and make it as efficient as possible.

REFRENCES:-

To conduct this project the following tools have been used :

- Pycharm Community Edition
- Tkinter (Library) : https://docs.python.org/3/library/tkinter.html
- Itertools (Library) : https://docs.python.org/2/library/itertools.html

1. Shedge Kishor N. , Sravan Kumar G : Solving Verbal Crypto-Arithmetic Problem by Parallel Genetic Algorithm(PGA), International Journal of Computer Technology and Electronics Engineering (IJCTEE) Volume 2, Issue 4,pp. 51-56, August 2012.

2. Hardik Soni & Nidhi Arora : Solving Crypt-Arithmetic Problems via Genetic Algorithm, JMIJIT, Volume-1 Issue 1, pp. 12-17, January, 2011, ISSN: Print 2229-6115.

3. Britannica : https://www.britannica.com/science/cryptarithm

4. Coursera : https://www.coursera.org/learn/introduction-to-ai