Artificial Intelligence  
Lab Exercise 3  
Constraint Satisfaction Problem (Cryptarithmetic)

short line

Ashwin Prakash  
RA1911026010048

**The Cryptarithmetic Problem:**

Cryptarithmetic Problem is a type of constraint satisfaction problem where the game is about digits and its unique replacement either with alphabets or other symbols.

In cryptarithmetic problem, the digits (0-9) get substituted by some possible alphabets or symbols. The task in cryptarithmetic problem is to substitute each digit with an alphabet to get the result arithmetically correct.

**Aim:**

To solve Cryptarithmetic Problem.

**Algorithm:**

**Rules:**

1. Each Letter, Symbol represents only one digit throughout the problem.
2. Numbers must not begin with zero i.e, should not be prefixed with zero (e.g: 0789 is wrong, 789 is correct).
3. Goal is to find the value of each letter in the Cryptatithmetic problems
4. There must only be one solution to the problem
5. The numerical base unless specified, is 10
6. After replacing letters by their digits, the resulting arithmetic operations must be correct.
7. Carry over can only be 1 in Cryptarithmetic problems

**Approach:**

1. Input the equation
2. Split the string on basis of equal signs into left and right parts
3. Add the right part and letters of the left part to a list
4. Permutate through mapping of digits (0-9) and list of letters
5. If the assigned permuted value is equal to the assigned right part value, that is a solution

**Program:**

import itertools

def assign\_value(word, sub):

    s = 0

    factor = 1

    for l in reversed(word):

        s += factor \* sub[l]

        factor \*= 10

    return s

def cryptsolve(eqn):

    lp, rp = eqn.lower().replace(' ','').split('=')

    lp = lp.split('+')

    letters = set(rp)

    for word in lp:

        for l in word:

            letters.add(l)

    letters = list(letters)

    digits = range(10)

    print('SOLUTIONS:')

    for perm in itertools.permutations(digits, len(letters)):

        soln = dict(zip(letters,perm))

        if sum(assign\_value(word, soln) for word in lp) == assign\_value(rp, soln):

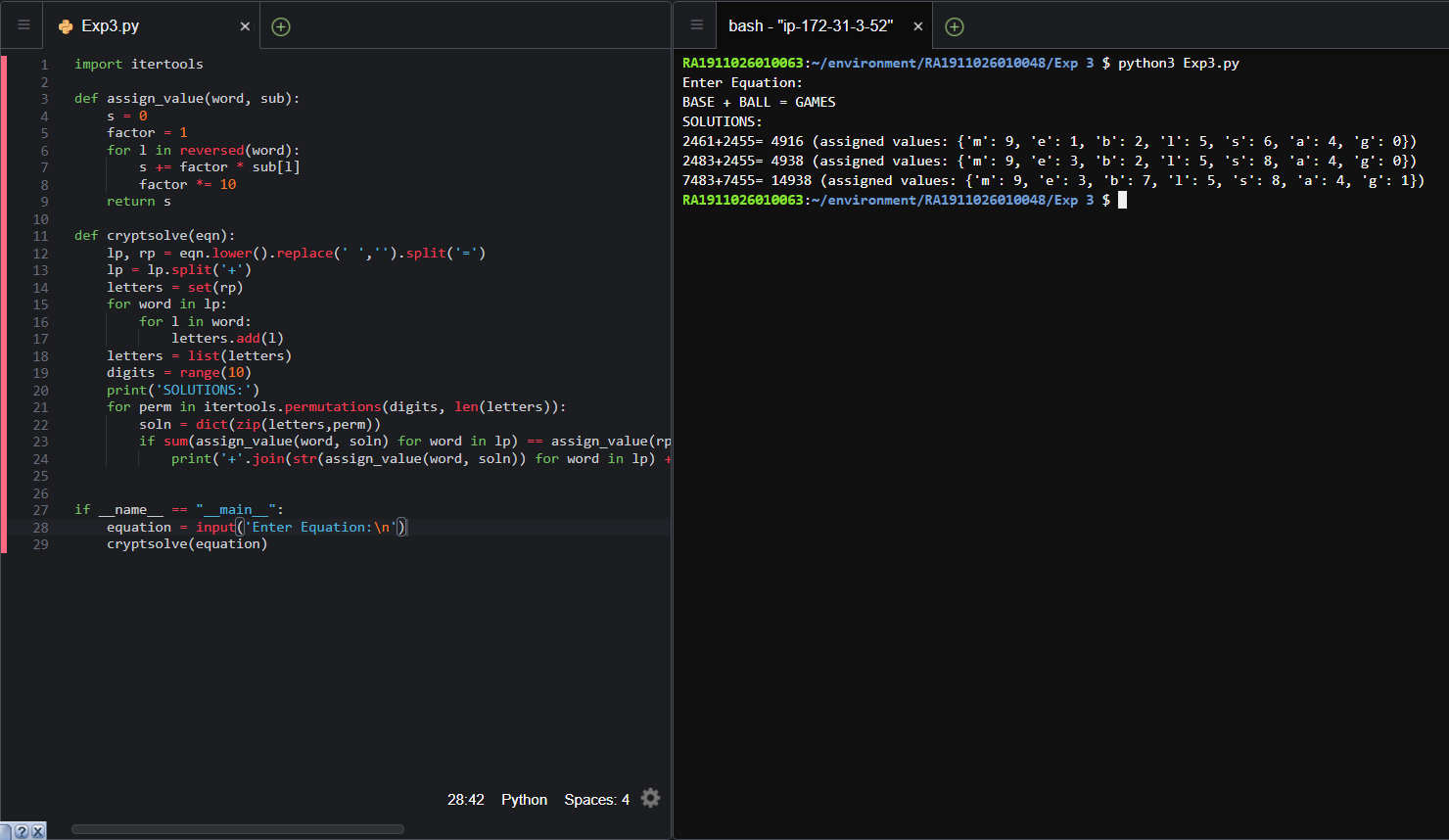
            print('+'.join(str(assign\_value(word, soln)) for word in lp) + f"= {assign\_value(rp, soln)} (assigned values: {soln})")

if \_\_name\_\_ == "\_\_main\_\_":

    equation = input()

    cryptsolve(equation)

**Output:**

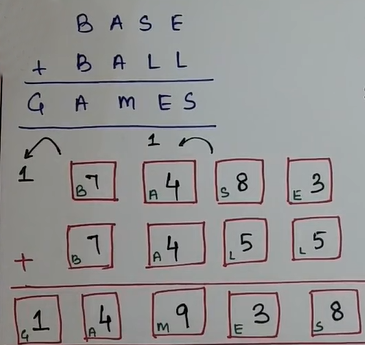
****

**Applications of Constraint Satisfaction Problem:**

* Map Coloring
* Word Search
* Eight Queens
* Magic Squares
* Conflict Resolution
* Sudoku Solver

**Observations:**

A hand-solved solution of the above input yields a result which is present in one of the output solutions of the program.



**Result:**

The Constraint Satisfaction Problem (Cryptarithmetic Problem) has been solved and verified by an example.