

AI LAB EXP-3

CONSTRAINT SATISFACTION PROBLEM

Date: 24-01-2022

Name: Vakada Siva Supradeep

Reg No: RA1911030010104

Github link: <https://github.com/Supradeepvakada/AI-LAB/tree/main/3.%20Constraint%20Satisfaction%20Problem>

CODE:

```
import itertools
```

```
def get_value(word, substitution):
```

```
    s = 0
```

```
    factor = 1
```

```
    for letter in reversed(word):
```

```
        s += factor * substitution[letter]
```

```
        factor *= 10
```

```
    return s
```

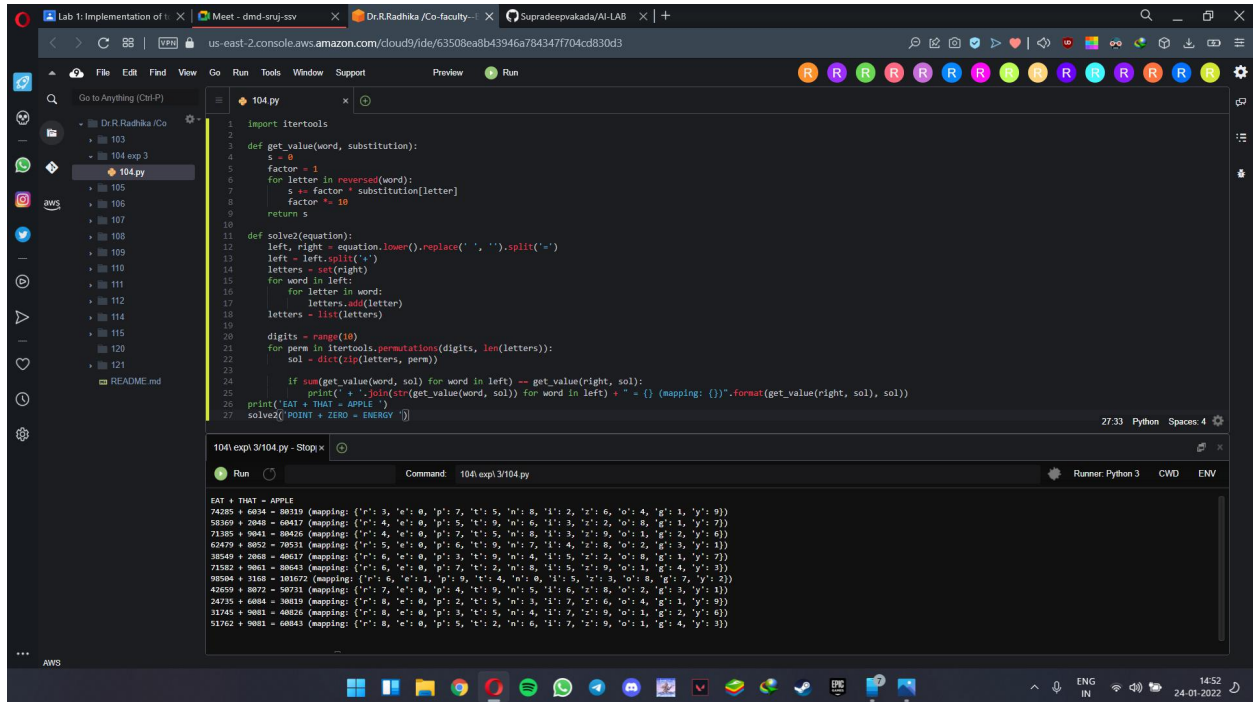
```
def solve2(equation):
```

```
left, right = equation.lower().replace(' ', '').split('=')
left = left.split('+')
letters = set(right)
for word in left:
    for letter in word:
        letters.add(letter)
letters = list(letters)

digits = range(10)
for perm in itertools.permutations(digits, len(letters)):
    sol = dict(zip(letters, perm))

    if sum(get_value(word, sol) for word in left) == get_value(right, sol):
        print(' + '.join(str(get_value(word, sol)) for word in left) + " = {}
(mapping: {})".format(get_value(right, sol), sol))
print('EAT + THAT = APPLE ')
solve2('POINT + ZERO = ENERGY ')
```

OUTPUT SCREENSHOT:



The screenshot displays a JupyterLab environment with a file explorer on the left, a code editor in the center, and an output console at the bottom. The code editor shows a Python script named '104.py' with the following content:

```
1 import itertools
2
3 def get_value(word, substitution):
4     s = 0
5     factor = 1
6     for letter in reversed(word):
7         s += factor * substitution[letter]
8         factor *= 10
9     return s
10
11 def solve2(equation):
12     left, right = equation.lower().replace(' ', '').split('=')
13     left = left.split('+')
14     letters = set(right)
15     for word in left:
16         for letter in word:
17             letters.add(letter)
18     letters = list(letters)
19
20     digits = range(10)
21     for perm in itertools.permutations(digits, len(letters)):
22         sol = dict(zip(letters, perm))
23
24         if sum(get_value(word, sol) for word in left) == get_value(right, sol):
25             print(" + ".join(str(get_value(word, sol)) for word in left) + " = {} (mapping: {})".format(get_value(right, sol), sol))
26
27 print("EAT + THAT = APPLE ")
28 solve2("EAT + THAT = APPLE ")
```

The output console shows the results of running the script, including the equation "EAT + THAT = APPLE" and a list of 10 mappings that satisfy the equation:

```
EAT + THAT = APPLE
74205 + 6034 = 80319 (mapping: {'r': 3, 'e': 0, 't': 7, 'a': 5, 'n': 8, 'i': 2, 'h': 6, 'o': 4, 'g': 1, 'y': 9})
58369 + 2048 = 60417 (mapping: {'r': 4, 'e': 0, 't': 5, 'a': 9, 'n': 6, 'i': 3, 'h': 2, 'o': 8, 'g': 1, 'y': 7})
71385 + 9041 = 80426 (mapping: {'r': 4, 'e': 0, 't': 7, 'a': 5, 'n': 8, 'i': 3, 'h': 2, 'o': 1, 'g': 1, 'y': 6})
62479 + 8052 = 70531 (mapping: {'r': 5, 'e': 0, 't': 6, 'a': 9, 'n': 7, 'i': 4, 'h': 2, 'o': 1, 'g': 1, 'y': 1})
38549 + 2048 = 40597 (mapping: {'r': 6, 'e': 0, 't': 3, 'a': 9, 'n': 4, 'i': 5, 'h': 2, 'o': 8, 'g': 1, 'y': 7})
71582 + 9041 = 80623 (mapping: {'r': 6, 'e': 0, 't': 7, 'a': 2, 'n': 8, 'i': 5, 'h': 9, 'o': 1, 'g': 4, 'y': 3})
90308 + 3180 = 101572 (mapping: {'r': 0, 'e': 1, 't': 9, 'a': 4, 'n': 0, 'i': 5, 'h': 3, 'o': 0, 'g': 7, 'y': 2})
42059 + 8072 = 50731 (mapping: {'r': 7, 'e': 0, 't': 4, 'a': 9, 'n': 5, 'i': 6, 'h': 8, 'o': 2, 'g': 1, 'y': 1})
24735 + 6084 = 30819 (mapping: {'r': 8, 'e': 0, 't': 2, 'a': 5, 'n': 3, 'i': 7, 'h': 6, 'o': 4, 'g': 1, 'y': 9})
31745 + 9041 = 40786 (mapping: {'r': 8, 'e': 0, 't': 3, 'a': 5, 'n': 4, 'i': 7, 'h': 9, 'o': 1, 'g': 2, 'y': 6})
51702 + 9041 = 60743 (mapping: {'r': 8, 'e': 0, 't': 5, 'a': 2, 'n': 6, 'i': 7, 'h': 9, 'o': 1, 'g': 4, 'y': 3})
```