In [ ]:
```python
import streamlit as st
from simpleai.search import CspProblem, backtrack

st.title("AI task part 1")
st.text("Type E.g: 'ODD + ODD = EVEN'. Press 'enter' to run the programme.")

# variables
input = input("Input: ")
originalList = []
uniqueList = []
beforePlus = []
beforeEquals = []
afterEquals = []
domains = {}

# put the input in a list using recursion
def inputToList(input):
    if (len(input)==0):
        return list
    else:
        if input[0] != ' ':
            originalList.append(input[0])
        return inputToList(input[1:])

# run the function
inputToList(input)

# put all the characters before the "+" in a seperate list
for char in originalList:
    if char != '+':
        beforePlus.append(char)
    else:
        break

# put all the characters after the "+" but before the "=" in a seperate list
for char in originalList[len(beforePlus)+1:]:
    if char != '=':
        beforeEquals.append(char)
    else:
        break

# put all the characters after the "=" in a seperate list
afterEquals = originalList[len(beforePlus) + 1 + len(beforeEquals) + 1:]

# put all the unique characters in a seperate list
for char in originalList:
    if char not in uniqueList and char != '+' and char != '=':
        uniqueList.append(char)

# set the unique list in a dictionary using recursion
def UniqueListToDictionary(uniqueList):
    if (len(uniqueList)==0):
        return domains
    else:
        # if the letter in the unique list matches the first letter of the word before "+",
        # before "=" and after "=" then range it from 1 to 10
        if uniqueList[0] == beforePlus[0] or uniqueList[0] == beforeEquals[0] or uniqueList[0] == afterEquals[0]:
            domains[uniqueList[0]] = list(range(1, 10))
        else:
            # otherwise range it from 0 to 10
            domains[uniqueList[0]] = list(range(0, 10))
        return UniqueListToDictionary(uniqueList[1:])

# run the function
UniqueListToDictionary(uniqueList)

# convert the uniqueList to a tupple called variables
variables = tuple(uniqueList)

def constraint_unique(variables, values):
    return len(values) == len(set(values))  # remove repeated values and count

def constraint_add(variables, values):
    # Convert the variables values into ints and create a dictionary
    # that maps each variable to its corresponding value.
    var_values = {variables[i]: values[i] for i in range(len(variables))}

    # Extract the left operand as a string by concatenating the values
    # of variables specified in the 'beforePlus' list.
    left_operand = ''.join([str(var_values[char]) for char in beforePlus])

    # Extract the right operand as a string by concatenating the values
    # of variables specified in the 'beforeEquals' list.
    right_operand = ''.join([str(var_values[char]) for char in beforeEquals])

    # Extract the expected result as a string by concatenating the values
    # of variables specified in the 'afterEquals' list.
    result = ''.join([str(var_values[char]) for char in afterEquals])

    # Check if the equation is valid by converting the operands and result
    # to integers and verifying if left_operand + right_operand equals result.
    return int(left_operand) + int(right_operand) == int(result)


constraints = [
    (variables, constraint_unique),
    (variables, constraint_add),
]

problem = CspProblem(variables, domains, constraints)

output = backtrack(problem)
print('\nSolutions:', output)
if output:
    st.text(output)
```

```
2023-10-05 09:43:05.635
  Warning: to view this Streamlit app on a browser, run it with the following
  command:

    streamlit run C:\Users\jarne\AppData\Roaming\Python\Python311\site-packages\ipykernel_launcher.py [ARGUMENTS]
Solutions: {'T': 2, 'O': 1, 'G': 8, 'U': 0}
```