Title: Implementation of Unification and Resolution

EX. NO : 07 Name: Parth Langalia

DATE : 14-03-2023 **Reg No. :** RA2011033010033

AIM :

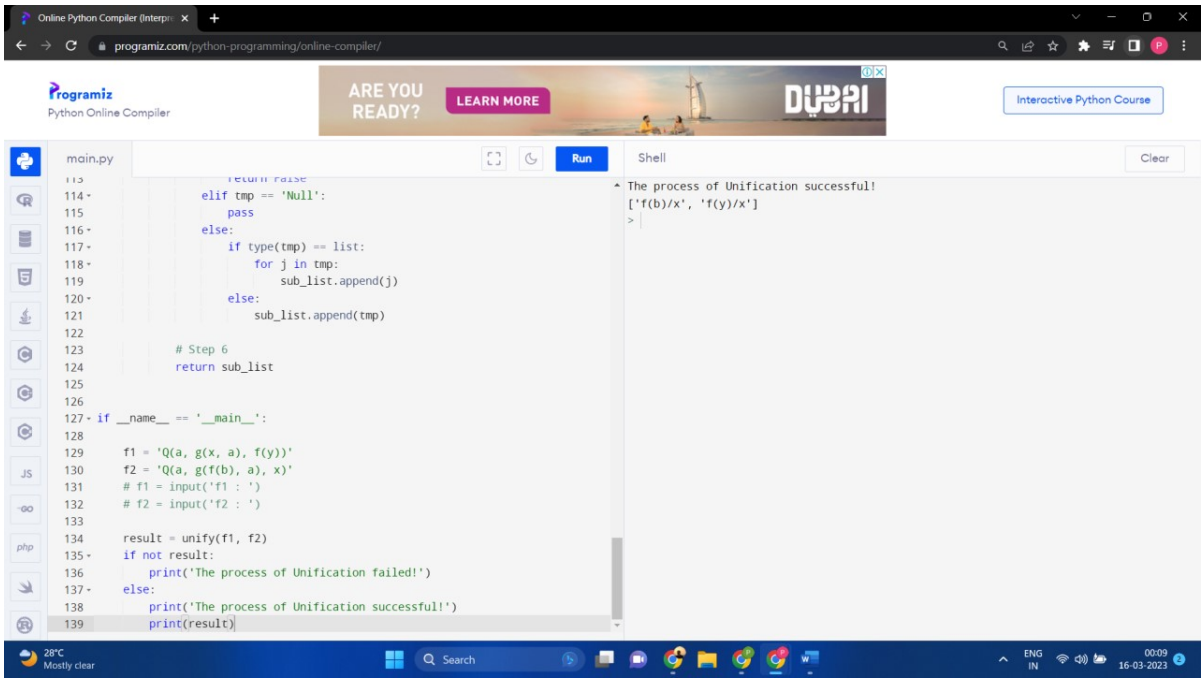
To Develop an optimized technique using an appropriate artificial intelligence algorithm to solve the Unification and Resolution

PSEUDO CODE :

1. function PL-RESOLUTION (KB, Q) returns true or false inputs: KB,
2. the knowledge base, group of sentences/facts in propositional logic
3. Q, the query, a sentence in propositional logic
4. clauses → the set of clauses in the CNF representation of KB ^ Q new → { }
5. loop do for each Ci, Cj in clauses do
6. resolvents → PL-RESOLVE (Ci, Cj)
7. if resolvents contains the empty clause the return true
8. new → new union resolvents
9. if new is a subset of clauses then return false
10. clauses → clauses union true

PROGRAM(with OUTPUT) :

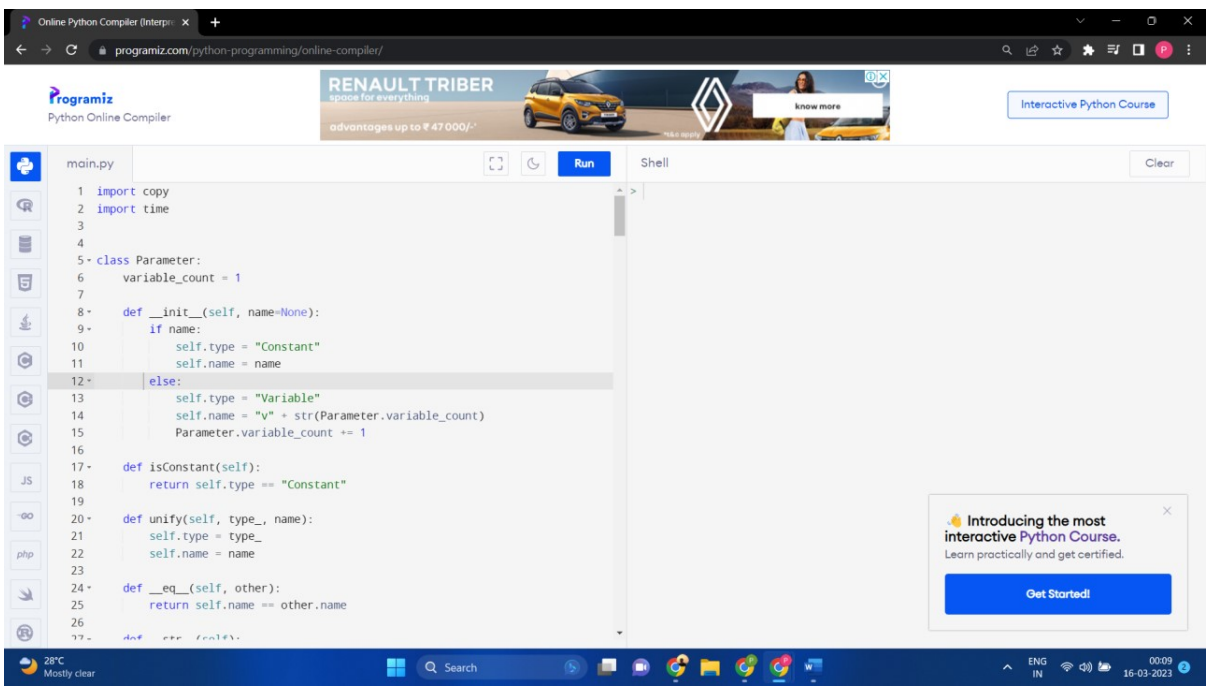
Unification:



```python
            elif tmp == 'Null':
                pass
            else:
                if type(tmp) == list:
                    for j in tmp:
                        sub_list.append(j)
                else:
                    sub_list.append(tmp)

        # Step 6
        return sub_list


if __name__ == '__main__':

    f1 = 'Q(a, g(x, a), f(y))'
    f2 = 'Q(a, g(f(b), a), x)'
    # f1 = input('f1 : ')
    # f2 = input('f2 : ')

    result = unify(f1, f2)
    if not result:
        print('The process of Unification failed!')
    else:
        print('The process of Unification successful!')
        print(result)
```

```
The process of Unification successful!
['f(b)/x', 'f(y)/x']
```

Resolution:



```python
import copy
import time


class Parameter:
    variable_count = 1

    def __init__(self, name=None):
        if name:
            self.type = "Constant"
            self.name = name
        else:
            self.type = "Variable"
            self.name = "v" + str(Parameter.variable_count)
            Parameter.variable_count += 1

    def isConstant(self):
        return self.type == "Constant"

    def unify(self, type_, name):
        self.type = type_
        self.name = name

    def __eq__(self, other):
        return self.name == other.name

    def __str__(self):
```

Manual Calculations :

Manual Calculation / Input/output :

Unification:

f1 : 'Q (a, g (x, a), f (y))'

f2 : 'Q (a, g (f(b), a), x)'

Process of unification successful:

['f(b) /x', 'f(y) /x']

Resolution :

Input:

6
A (Alice)
¬A (Alice)
Z (zig)
¬Z (zig)

G (Golf)
¬G (Good)

10

A(x) ⇒ B(x)
B(x) ⇒ C(x)
C(x) ⇒ D(x)
D(x) ⇒ E(x)
E(x) ⇒ A(x)

G (g)

H (h)

I (i)

J (j)

K (k)

Output:

['FALSE', 'FALSE', 'FALSE', 'FALSE', 'TRUE', 'FALSE'].

RESULT :

Developed Unification and Resolution Algorithm in Python for solving logical problems.