Aidan Kiser

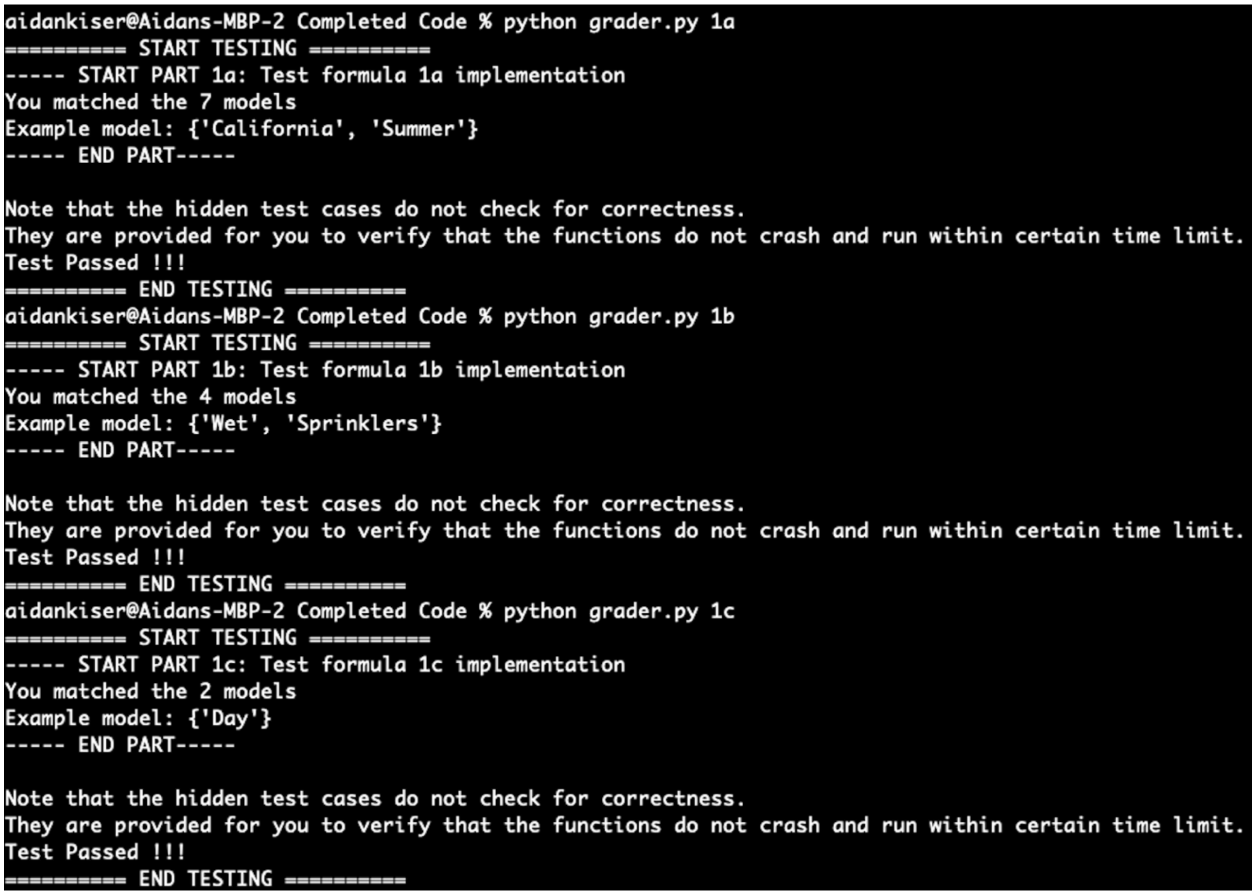
13 February 2024

COMP-5600

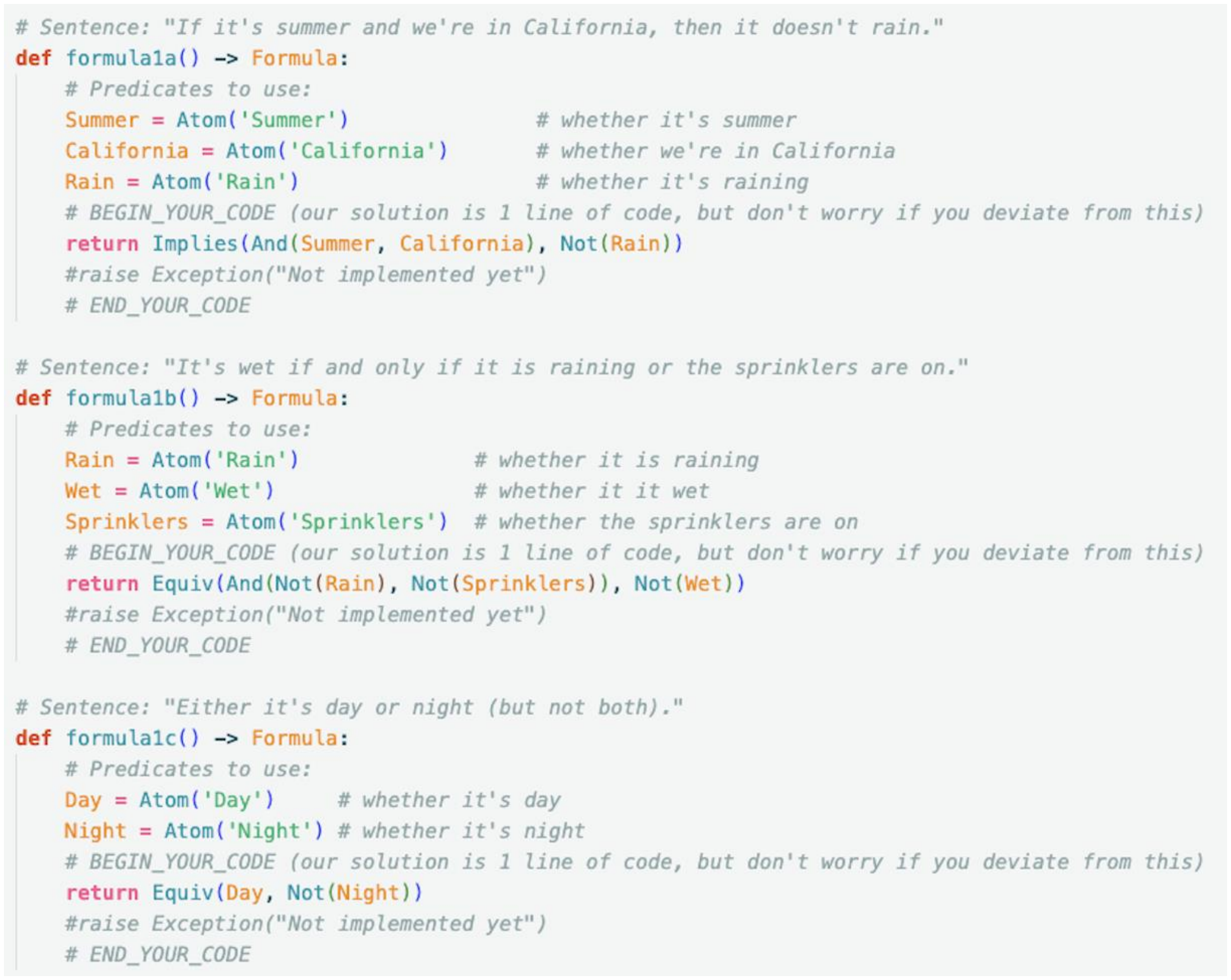
Assignment 2: Logic & Semantics

Screenshots

Number 1:

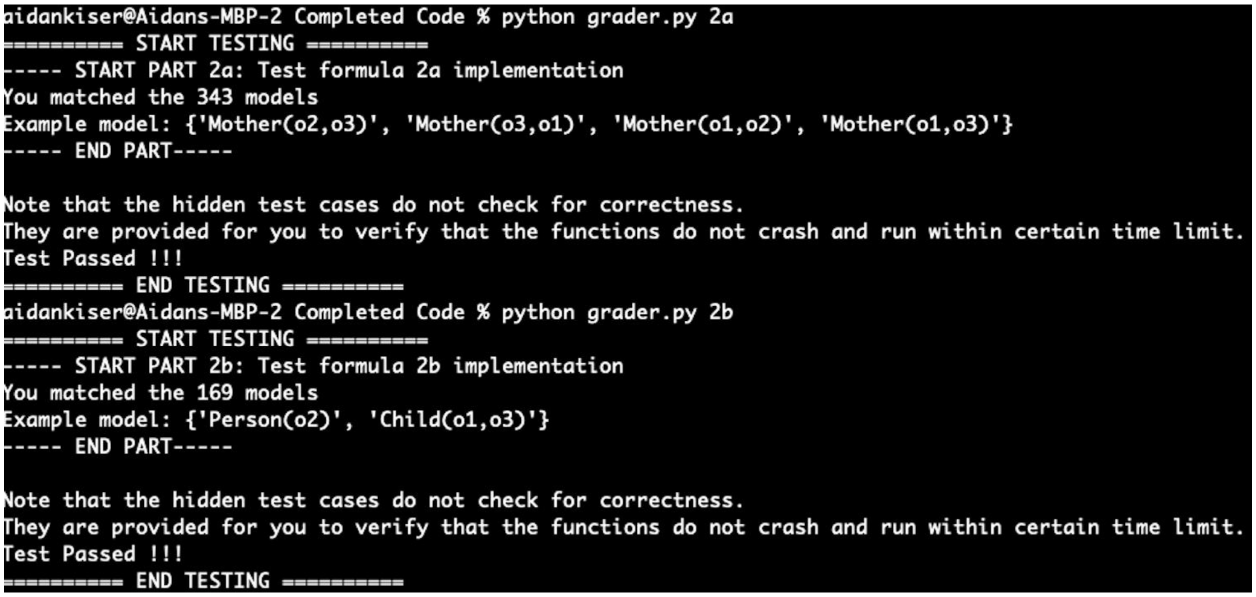


```
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 1a
========== START TESTING ==========
----- START PART 1a: Test formula 1a implementation
You matched the 7 models
Example model: {'California', 'Summer'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 1b
========== START TESTING ==========
----- START PART 1b: Test formula 1b implementation
You matched the 4 models
Example model: {'Wet', 'Sprinklers'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 1c
========== START TESTING ==========
----- START PART 1c: Test formula 1c implementation
You matched the 2 models
Example model: {'Day'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
```

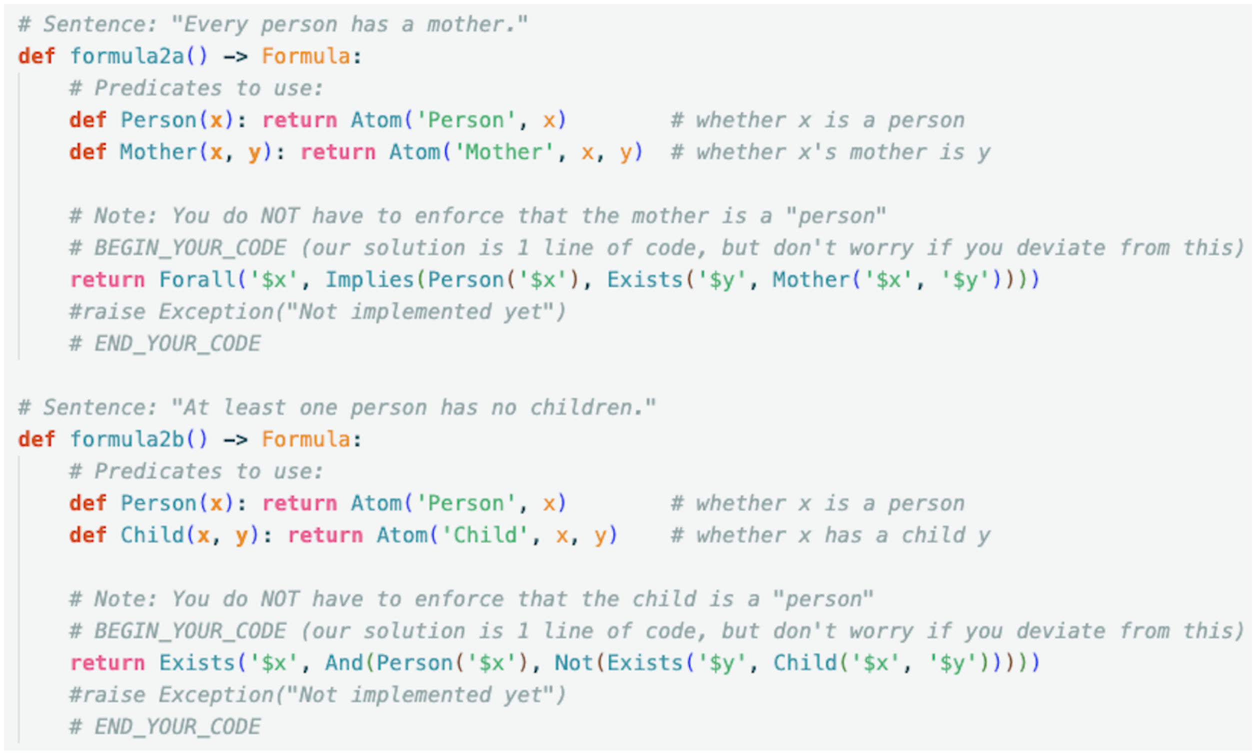


```python
# Sentence: "If it's summer and we're in California, then it doesn't rain."
def formula1a() -> Formula:
    # Predicates to use:
    Summer = Atom('Summer')               # whether it's summer
    California = Atom('California')       # whether we're in California
    Rain = Atom('Rain')                   # whether it's raining
    # BEGIN_YOUR_CODE (our solution is 1 line of code, but don't worry if you deviate from this)
    return Implies(And(Summer, California), Not(Rain))
    #raise Exception("Not implemented yet")
    # END_YOUR_CODE

# Sentence: "It's wet if and only if it is raining or the sprinklers are on."
def formula1b() -> Formula:
    # Predicates to use:
    Rain = Atom('Rain')              # whether it is raining
    Wet = Atom('Wet')                # whether it it wet
    Sprinklers = Atom('Sprinklers')  # whether the sprinklers are on
    # BEGIN_YOUR_CODE (our solution is 1 line of code, but don't worry if you deviate from this)
    return Equiv(And(Not(Rain), Not(Sprinklers)), Not(Wet))
    #raise Exception("Not implemented yet")
    # END_YOUR_CODE

# Sentence: "Either it's day or night (but not both)."
def formula1c() -> Formula:
    # Predicates to use:
    Day = Atom('Day')     # whether it's day
    Night = Atom('Night') # whether it's night
    # BEGIN_YOUR_CODE (our solution is 1 line of code, but don't worry if you deviate from this)
    return Equiv(Day, Not(Night))
    #raise Exception("Not implemented yet")
    # END_YOUR_CODE
```

Number 2:

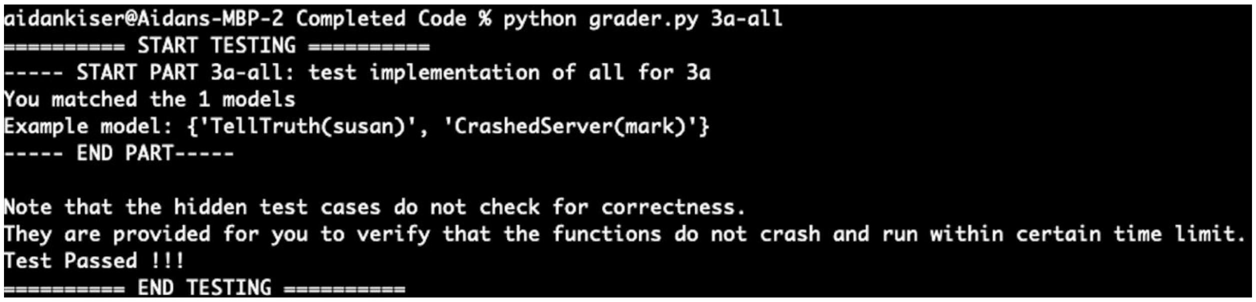


```
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 2a
========== START TESTING ==========
----- START PART 2a: Test formula 2a implementation
You matched the 343 models
Example model: {'Mother(o2,o3)', 'Mother(o3,o1)', 'Mother(o1,o2)', 'Mother(o1,o3)'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 2b
========== START TESTING ==========
----- START PART 2b: Test formula 2b implementation
You matched the 169 models
Example model: {'Person(o2)', 'Child(o1,o3)'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
```

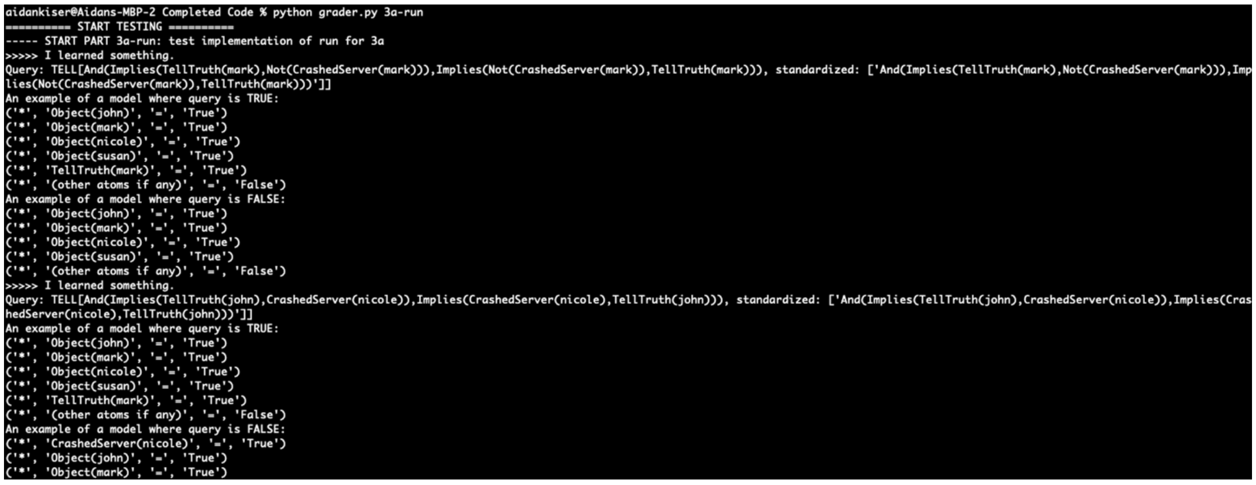


```python
# Sentence: "Every person has a mother."
def formula2a() -> Formula:
    # Predicates to use:
    def Person(x): return Atom('Person', x)        # whether x is a person
    def Mother(x, y): return Atom('Mother', x, y)  # whether x's mother is y

    # Note: You do NOT have to enforce that the mother is a "person"
    # BEGIN_YOUR_CODE (our solution is 1 line of code, but don't worry if you deviate from this)
    return Forall('$x', Implies(Person('$x'), Exists('$y', Mother('$x', '$y'))))
    #raise Exception("Not implemented yet")
    # END_YOUR_CODE

# Sentence: "At least one person has no children."
def formula2b() -> Formula:
    # Predicates to use:
    def Person(x): return Atom('Person', x)        # whether x is a person
    def Child(x, y): return Atom('Child', x, y)    # whether x has a child y

    # Note: You do NOT have to enforce that the child is a "person"
    # BEGIN_YOUR_CODE (our solution is 1 line of code, but don't worry if you deviate from this)
    return Exists('$x', And(Person('$x'), Not(Exists('$y', Child('$x', '$y')))))
    #raise Exception("Not implemented yet")
    # END_YOUR_CODE
```

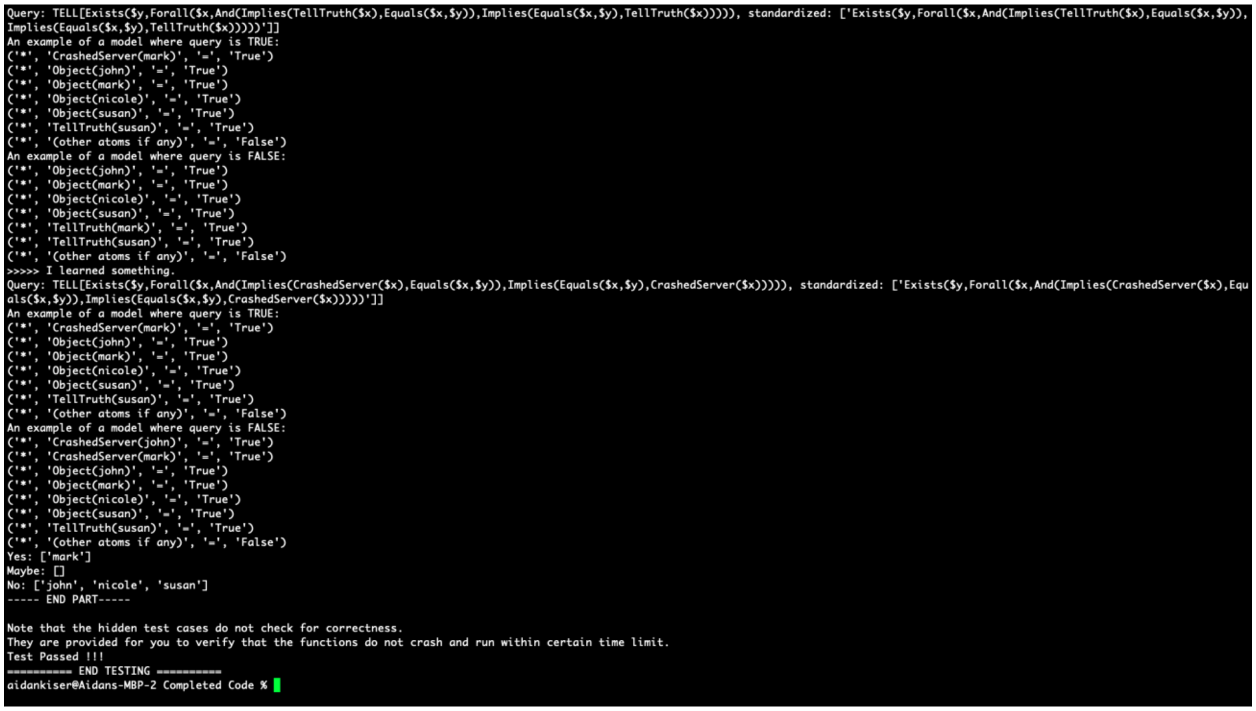
Number 3:



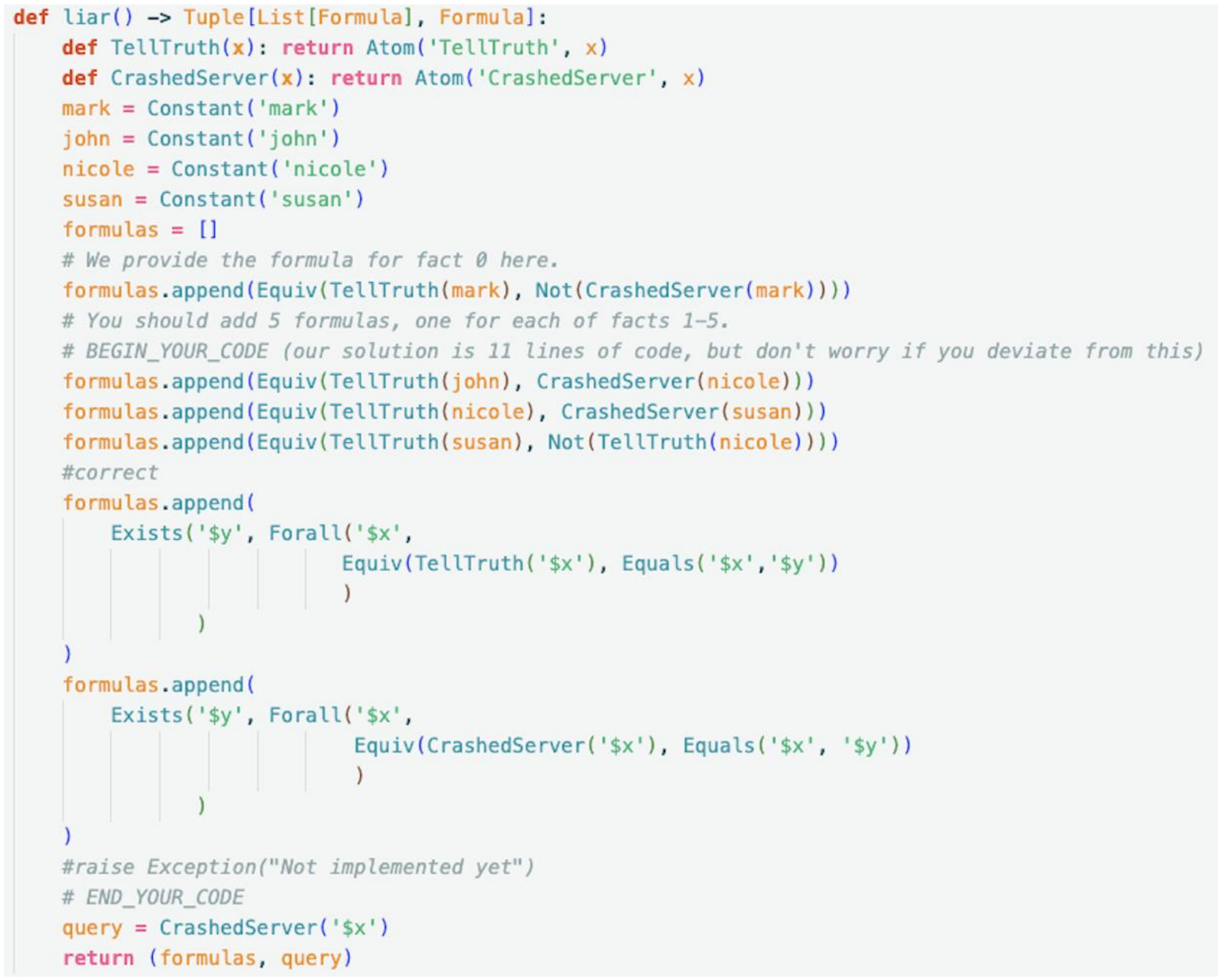
```
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 3a-all
========== START TESTING ==========
----- START PART 3a-all: test implementation of all for 3a
You matched the 1 models
Example model: {'TellTruth(susan)', 'CrashedServer(mark)'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
```



```
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 3a-run
========== START TESTING ==========
----- START PART 3a-run: test implementation of run for 3a
>>>>> I learned something.
Query: TELL[And(Implies(TellTruth(mark),Not(CrashedServer(mark))),Implies(Not(CrashedServer(mark)),TellTruth(mark))), standardized: ['And(Implies(TellTruth(mark),Not(CrashedServer(mark))),Imp
lies(Not(CrashedServer(mark)),TellTruth(mark)))']]
An example of a model where query is TRUE:
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', 'TellTruth(mark)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
>>>>> I learned something.
Query: TELL[And(Implies(TellTruth(john),CrashedServer(nicole)),Implies(CrashedServer(nicole),TellTruth(john))), standardized: ['And(Implies(TellTruth(john),CrashedServer(nicole)),Implies(Cras
hedServer(nicole),TellTruth(john)))']]
An example of a model where query is TRUE:
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', 'TellTruth(mark)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'CrashedServer(nicole)', '=', 'True')
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
```

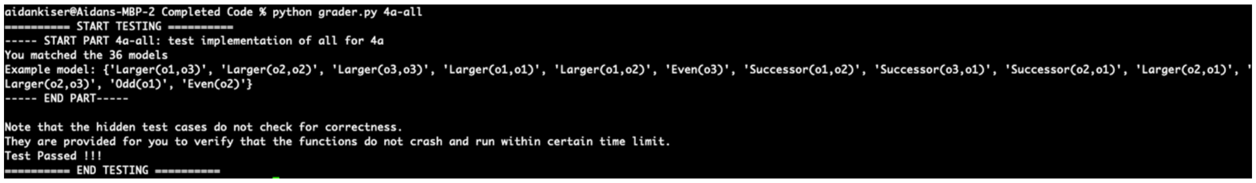


```
Query: TELL[Exists($y,Forall($x,And(Implies(TellTruth($x),Equals($x,$y)),Implies(Equals($x,$y),TellTruth($x))))), standardized: ['Exists($y,Forall($x,And(Implies(TellTruth($x),Equals($x,$y)),
Implies(Equals($x,$y),TellTruth($x)))))']]
An example of a model where query is TRUE:
('*', 'CrashedServer(mark)', '=', 'True')
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', 'TellTruth(susan)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', 'TellTruth(mark)', '=', 'True')
('*', 'TellTruth(susan)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
>>>>> I learned something.
Query: TELL[Exists($y,Forall($x,And(Implies(CrashedServer($x),Equals($x,$y)),Implies(Equals($x,$y),CrashedServer($x))))), standardized: ['Exists($y,Forall($x,And(Implies(CrashedServer($x),Equ
als($x,$y)),Implies(Equals($x,$y),CrashedServer($x)))))']]
An example of a model where query is TRUE:
('*', 'CrashedServer(mark)', '=', 'True')
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', 'TellTruth(susan)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'CrashedServer(john)', '=', 'True')
('*', 'CrashedServer(mark)', '=', 'True')
('*', 'Object(john)', '=', 'True')
('*', 'Object(mark)', '=', 'True')
('*', 'Object(nicole)', '=', 'True')
('*', 'Object(susan)', '=', 'True')
('*', 'TellTruth(susan)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
Yes: ['mark']
Maybe: []
No: ['john', 'nicole', 'susan']
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
aidankiser@Aidans-MBP-2 Completed Code %
```

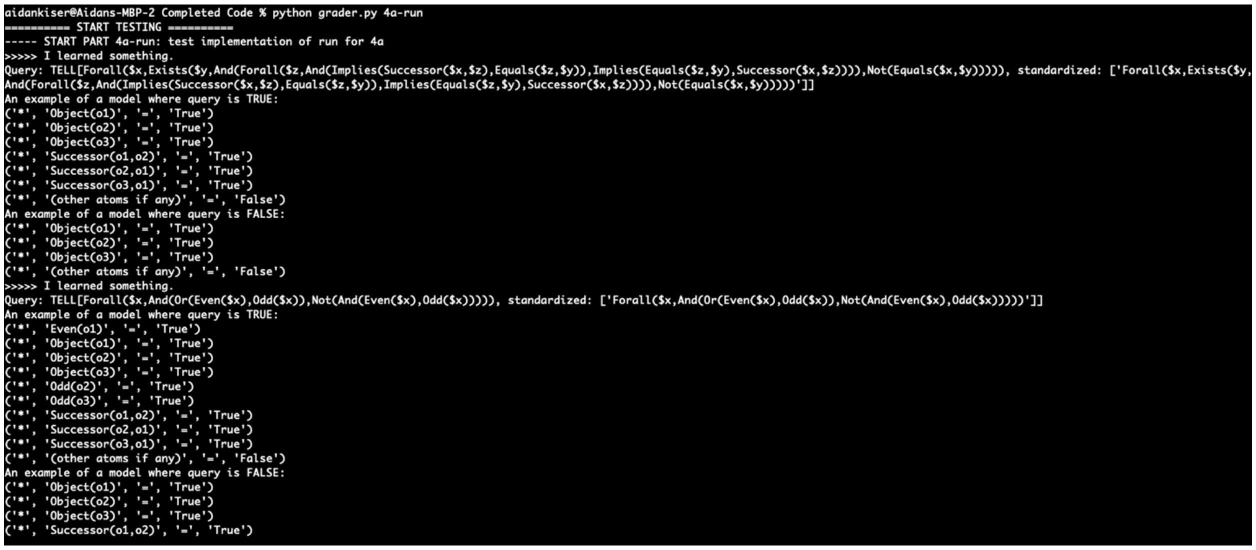


```python
def liar() -> Tuple[List[Formula], Formula]:
    def TellTruth(x): return Atom('TellTruth', x)
    def CrashedServer(x): return Atom('CrashedServer', x)
    mark = Constant('mark')
    john = Constant('john')
    nicole = Constant('nicole')
    susan = Constant('susan')
    formulas = []
    # We provide the formula for fact 0 here.
    formulas.append(Equiv(TellTruth(mark), Not(CrashedServer(mark))))
    # You should add 5 formulas, one for each of facts 1-5.
    # BEGIN_YOUR_CODE (our solution is 11 lines of code, but don't worry if you deviate from this)
    formulas.append(Equiv(TellTruth(john), CrashedServer(nicole)))
    formulas.append(Equiv(TellTruth(nicole), CrashedServer(susan)))
    formulas.append(Equiv(TellTruth(susan), Not(TellTruth(nicole))))
    #correct
    formulas.append(
        Exists('$y', Forall('$x',
                                Equiv(TellTruth('$x'), Equals('$x','$y'))
                                )
                    )
    )
    formulas.append(
        Exists('$y', Forall('$x',
                                 Equiv(CrashedServer('$x'), Equals('$x', '$y'))
                                 )
                    )
    )
    #raise Exception("Not implemented yet")
    # END_YOUR_CODE
    query = CrashedServer('$x')
    return (formulas, query)
```

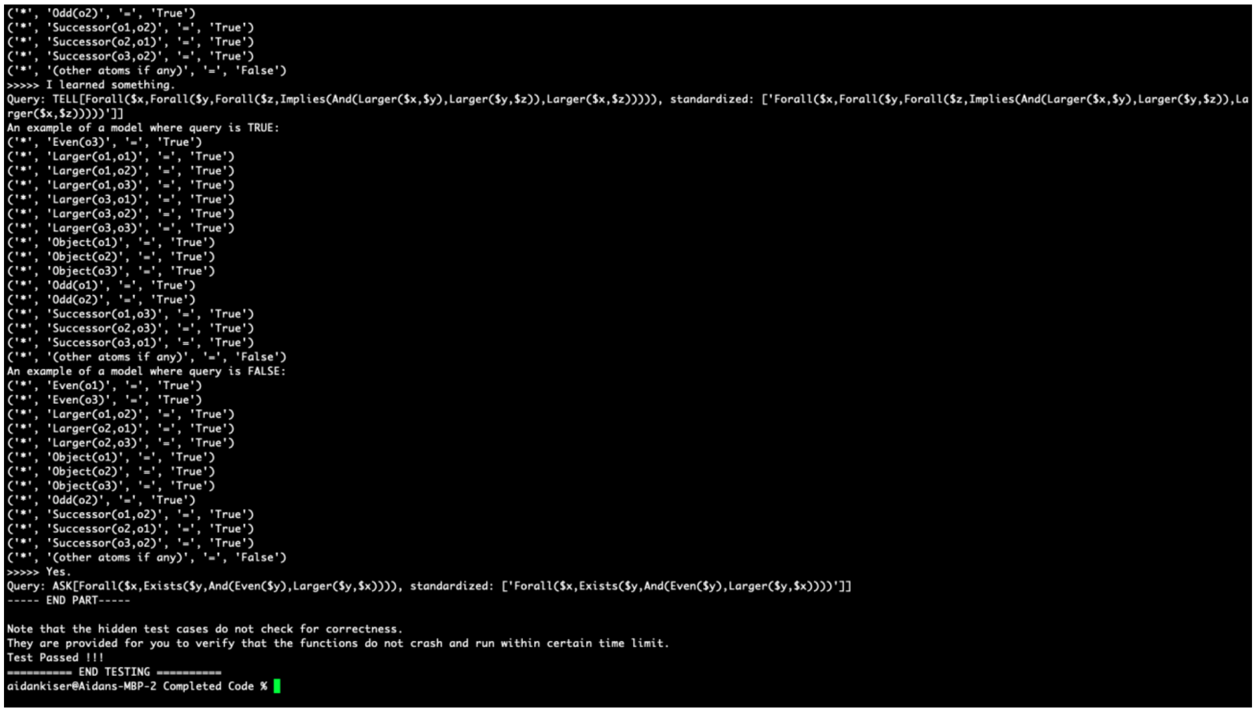
Number 4:



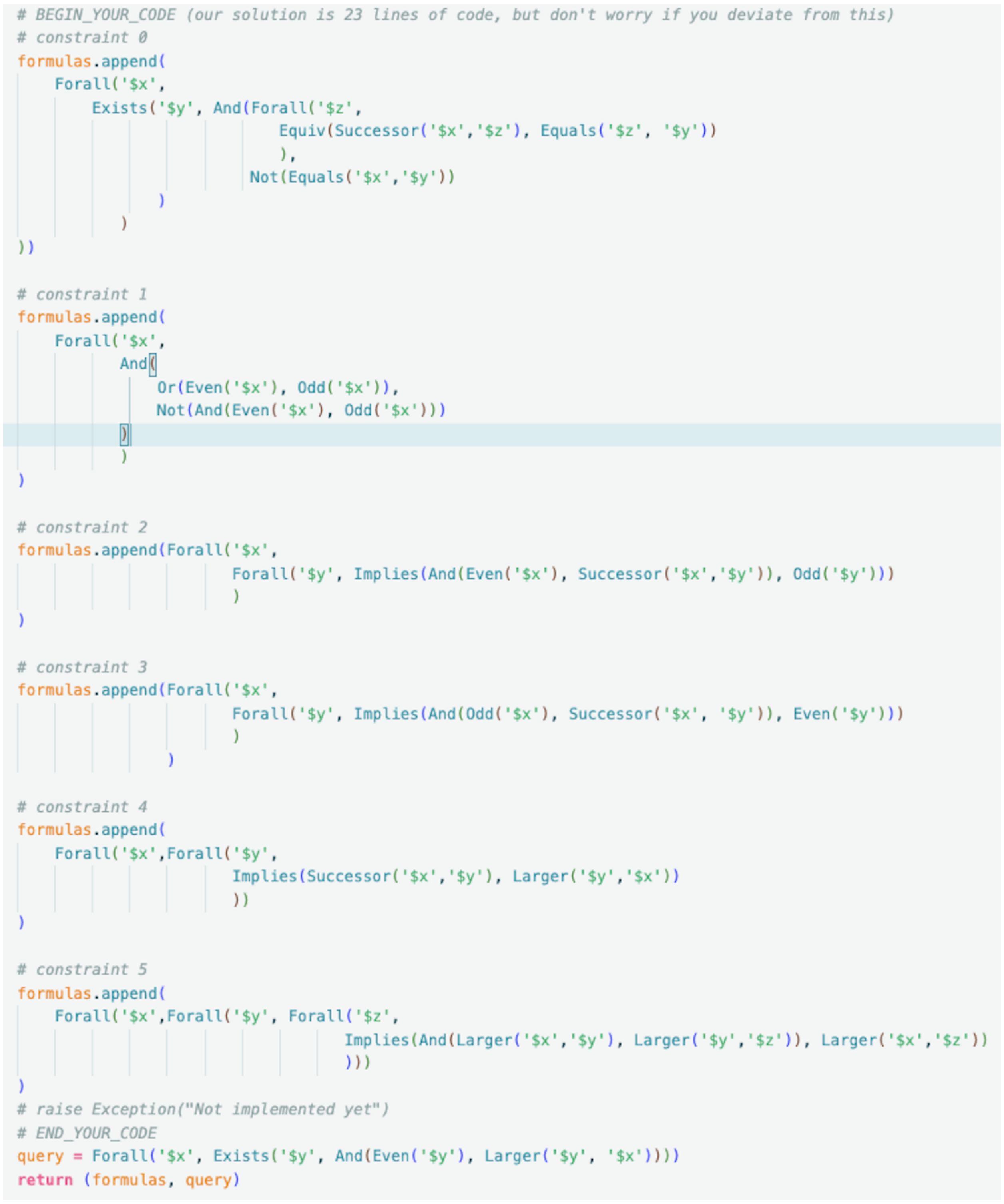
```
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 4a-all
========== START TESTING ==========
----- START PART 4a-all: test implementation of all for 4a
You matched the 36 models
Example model: {'Larger(o1,o3)', 'Larger(o2,o2)', 'Larger(o3,o3)', 'Larger(o1,o1)', 'Larger(o1,o2)', 'Even(o3)', 'Successor(o1,o2)', 'Successor(o3,o1)', 'Successor(o2,o1)', 'Larger(o2,o1)', '
Larger(o2,o3)', 'Odd(o1)', 'Even(o2)'}
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
```



```
aidankiser@Aidans-MBP-2 Completed Code % python grader.py 4a-run
========== START TESTING ==========
----- START PART 4a-run: test implementation of run for 4a
>>>>> I learned something.
Query: TELL[Forall($x,Exists($y,And(Forall($z,And(Implies(Successor($x,$z),Equals($z,$y)),Implies(Equals($z,$y),Successor($x,$z)))),Not(Equals($x,$y))))), standardized: ['Forall($x,Exists($y,
And(Forall($z,And(Implies(Successor($x,$z),Equals($z,$y)),Implies(Equals($z,$y),Successor($x,$z)))),Not(Equals($x,$y)))))']]
An example of a model where query is TRUE:
('*', 'Object(o1)', '=', 'True')
('*', 'Object(o2)', '=', 'True')
('*', 'Object(o3)', '=', 'True')
('*', 'Successor(o1,o2)', '=', 'True')
('*', 'Successor(o2,o1)', '=', 'True')
('*', 'Successor(o3,o1)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'Object(o1)', '=', 'True')
('*', 'Object(o2)', '=', 'True')
('*', 'Object(o3)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
>>>>> I learned something.
Query: TELL[Forall($x,And(Or(Even($x),Odd($x)),Not(And(Even($x),Odd($x))))), standardized: ['Forall($x,And(Or(Even($x),Odd($x)),Not(And(Even($x),Odd($x)))))']]
An example of a model where query is TRUE:
('*', 'Even(o1)', '=', 'True')
('*', 'Object(o1)', '=', 'True')
('*', 'Object(o2)', '=', 'True')
('*', 'Object(o3)', '=', 'True')
('*', 'Odd(o2)', '=', 'True')
('*', 'Odd(o3)', '=', 'True')
('*', 'Successor(o1,o2)', '=', 'True')
('*', 'Successor(o2,o1)', '=', 'True')
('*', 'Successor(o3,o1)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'Object(o1)', '=', 'True')
('*', 'Object(o2)', '=', 'True')
('*', 'Object(o3)', '=', 'True')
('*', 'Successor(o1,o2)', '=', 'True')
```



```
('*', 'Odd(o2)', '=', 'True')
('*', 'Successor(o1,o2)', '=', 'True')
('*', 'Successor(o2,o1)', '=', 'True')
('*', 'Successor(o3,o2)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
>>>>> I learned something.
Query: TELL[Forall($x,Forall($y,Forall($z,Implies(And(Larger($x,$y),Larger($y,$z)),Larger($x,$z))))), standardized: ['Forall($x,Forall($y,Forall($z,Implies(And(Larger($x,$y),Larger($y,$z)),La
rger($x,$z)))))']]
An example of a model where query is TRUE:
('*', 'Even(o3)', '=', 'True')
('*', 'Larger(o1,o1)', '=', 'True')
('*', 'Larger(o1,o2)', '=', 'True')
('*', 'Larger(o1,o3)', '=', 'True')
('*', 'Larger(o3,o1)', '=', 'True')
('*', 'Larger(o3,o2)', '=', 'True')
('*', 'Larger(o3,o3)', '=', 'True')
('*', 'Object(o1)', '=', 'True')
('*', 'Object(o2)', '=', 'True')
('*', 'Object(o3)', '=', 'True')
('*', 'Odd(o1)', '=', 'True')
('*', 'Odd(o2)', '=', 'True')
('*', 'Successor(o1,o3)', '=', 'True')
('*', 'Successor(o2,o3)', '=', 'True')
('*', 'Successor(o3,o1)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
An example of a model where query is FALSE:
('*', 'Even(o1)', '=', 'True')
('*', 'Even(o3)', '=', 'True')
('*', 'Larger(o1,o2)', '=', 'True')
('*', 'Larger(o2,o1)', '=', 'True')
('*', 'Larger(o2,o3)', '=', 'True')
('*', 'Object(o1)', '=', 'True')
('*', 'Object(o2)', '=', 'True')
('*', 'Object(o3)', '=', 'True')
('*', 'Odd(o2)', '=', 'True')
('*', 'Successor(o1,o2)', '=', 'True')
('*', 'Successor(o2,o1)', '=', 'True')
('*', 'Successor(o3,o2)', '=', 'True')
('*', '(other atoms if any)', '=', 'False')
>>>>> Yes.
Query: ASK[Forall($x,Exists($y,And(Even($y),Larger($y,$x)))), standardized: ['Forall($x,Exists($y,And(Even($y),Larger($y,$x))))']]
----- END PART-----

Note that the hidden test cases do not check for correctness.
They are provided for you to verify that the functions do not crash and run within certain time limit.
Test Passed !!!
========== END TESTING ==========
aidankiser@Aidans-MBP-2 Completed Code %
```



```python
# BEGIN_YOUR_CODE (our solution is 23 lines of code, but don't worry if you deviate from this)
# constraint 0
formulas.append(
    Forall('$x',
        Exists('$y', And(Forall('$z',
                            Equiv(Successor('$x','$z'), Equals('$z', '$y'))
                            ),
                         Not(Equals('$x','$y'))
               )
           )
))

# constraint 1
formulas.append(
    Forall('$x',
           And(
               Or(Even('$x'), Odd('$x')),
               Not(And(Even('$x'), Odd('$x')))
           )
           )
)

# constraint 2
formulas.append(Forall('$x',
                       Forall('$y', Implies(And(Even('$x'), Successor('$x','$y')), Odd('$y')))
                       )
)

# constraint 3
formulas.append(Forall('$x',
                       Forall('$y', Implies(And(Odd('$x'), Successor('$x', '$y')), Even('$y')))
                       )
                )

# constraint 4
formulas.append(
    Forall('$x',Forall('$y',
                       Implies(Successor('$x','$y'), Larger('$y','$x'))
                       ))
)

# constraint 5
formulas.append(
    Forall('$x',Forall('$y', Forall('$z',
                                   Implies(And(Larger('$x','$y'), Larger('$y','$z')), Larger('$x','$z'))
                                   )))
)
# raise Exception("Not implemented yet")
# END_YOUR_CODE
query = Forall('$x', Exists('$y', And(Even('$y'), Larger('$y', '$x'))))
return (formulas, query)
```