Asignment 06

Overview

So, I did things slightly different than directed, but in a way I'm pretty sure still satisfies the requirements.

Implementation Structure

Each implementation has 5-6 main parts:

- A problem Generator That Will Randomly Generate a Problem of Size n
- A Recursive Solution
- A Memoizing Solution
- A DP Solution
- A Traceback for the DP Solution
- A Printer for the Traceback

Implementation code can be found in the *implementations.py* file.

Testing Structure

The testing relies on 2-3 main functions.

Test Equal

This fucntion tests the recursive, memoizing, and DP solutions on a set of problems of varying sizes. It starts with a given size, and ends with a given size, going in increments of 2. It uses the problem generator mentioned above to generate a set of problems at the current size, then makes sure each algorithm returns the same result for each problem in the set. This provides a much more rigerous testing environment than requested, and should match the requirment for "four smaller distinct problems that the recursive solution returns the same answer as the DP", as our randomly generated problems are pretty much gaurenteed to be unique. A side note is that neither this function, nor any other part of my testing code prints the problems to the screen. That's because these problems get very large, and aren't printable very easily. Most of the problem information should be included in the traceback printer anyways.

Most of these recursive algorithms are very inefficient. That means that large problem sizes aren't feasibly testible for them.

It should be noted that for generalization purposes, each generator returns an unpackable object.

Test DP

This function tests the DP on a set of known problems and their solutions. It makes sure that for each problem, the DP generates the correct result, that's been determined beforehand.

Traceback/Print Traceback

These functions are implemented for each problem as mentioned above. After the Test DP function is called on each known problem, the traceback/print traceback is used to print the results to the console. This should satisfy the requirment "show the traceback routine working for four diverse of problems", whatever that actually means.

Overall Flow

So basically, we use the random problem generator to create a large quantity of varying sizes of random problems. We then test each implementation on these problems to ensure their equal.

Once we're sure they all return the same results, we then test the DP algorithm on serveral "KNOWN" problems. Since all algorithms are shown to be equal above, if the DP works, then the other algorithms should work as well.

Finally, we run some more known problems that produce valid results through the traceback, to see how the results are actually devised.

Testing code can be found in the *tests.py* file.

Substring Problem

Known Problems:

```
Problem 0:
S: catcatadogpersonctdogolargecatdog"
Words: ["cat", "dog", "person", "ct", "cata", "dogo", "large"]
result: True

Problem 1:
S: "ctctctctctctctctctcta"
Words: ["cat", "dog", "person", "ct", "cata", "dogo", "large"]
result: False

Problem 2:
S: "abababbbbababbabaab"
Words: ["abaab", "babba", "aab", "bba"]
Result: False

Problem 3:
S: "isfhasodihgoaspihgosad"
Words: ["bba"]
result: False
```

Traceback Problems:

```
Problem 0:
S: "catcatadogpersonctdogolargecatdog"
Words: ["cat", "dog", "person", "ct", "cata", "dogo", "large"]

Problem 1:
S: "abababababababaa"
Words: ["ab", "aa"]
```

```
Problem 2:
S: "sicksoulsoupstonesalarycatsctssicksalary"
Words: ["sick", "salary", "soul", "soup", "stone", "cats", "cts"]

Problem 3:
S: "imbluedabbadeedabbadie"
Words: ["die", "dabba", "dee", "blue", "im"]
```

Equality Results:

```
----------Equality Test----------
Testing Problem Size 8
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
Testing Problem Size 16
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
```

```
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
```

DP Results:

```
----------DP Test----------
Testing DP Algorithm:
    Expected: True, Result: True -> Pass
    Expected: False, Result: False -> Pass
    Expected: False, Result: False -> Pass
    Expected: False, Result: False -> Pass
```

Traceback:

```
----------TraceBack Test----------
Problem 0:
WORDS:
['cat', 'dog', 'person', 'ct', 'cata', 'dogo', 'large']
ORIGINAL STRING:
catcatadogpersonctdogolargecatdog
cat | cata | dog | person | ct | dogo | large | cat | dog

Problem 1:
WORDS:
['ab', 'aa']
ORIGINAL STRING:
abababababaa
ab | ab | ab | ab | ab | ab | aa

Problem 2:
WORDS:
['sick', 'salary', 'soul', 'soup', 'stone', 'cats', 'cts']
ORIGINAL STRING:
sicksoulsoupstonesalarycatsctssicksalary
sick | soul | soup | stone | salary | cats | cts | sick | salary

Problem 3:
WORDS:
```

```
['die', 'dabba', 'dee', 'blue', 'im']
ORIGINAL STRING:
imbluedabbadeedabbadie
im | blue | dabba | dee | dabba | die
```

3D Grid Problem (2)

This entire problem is pretty hard to show results for, so I've only included what's reasonable.

Known Problems:

```
Problem 0:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(1, 0, 0), (0, 1, 0), (0, 0, 1)]

Problem 1:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(0, 0, 0)]

Problem 2:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(1, 1, 1)]

Problem 3:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(0, 1, 0), (0, 0, 1)]
```

Traceback Problems:

```
Problem 0:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(1, 0, 0), (0, 1, 0), (0, 0, 1)]

Problem 1:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(0, 0, 0)]

Problem 2:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(1, 1, 1)]

Problem 3:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Known Moves: [(0, 1, 0), (0, 0, 1)]
```

Equality Results:

```
----------Equality Test----------
Testing Problem Size 16
    Problem 0-> Pass
    Problem 1-> Pass
    Problem 2-> Pass
    Problem 3-> Pass
Testing Problem Size 32
    Problem 0-> Pass
    Problem 1-> Pass
    Problem 2-> Pass
    Problem 3-> Pass
```

DP Results:

```
----------DP Test----------
Testing DP Algorithm:
    Expected: [[[True, True, True], [True, True, True], [True, True, True]],
[[True, True, True], [True, True, True], [True, True, True]], [[True, True,
True], [True, True, True], [True, True, True]]]
    Result: [[[(True, (-1, -1, -1)), (True, (0, 0, 0)), (True, (0, 0, 1))],
[(True, (0, 0, 0)), (True, (0, 1, 0)), (True, (0, 1, 1))], [(True, (0, 1, 0)),
(True, (0, 2, 0)), (True, (0, 2, 1))]], [[(True, (0, 0, 0)), (True, (1, 0, 0)),
(True, (1, 0, 1))], [(True, (1, 0, 0)), (True, (1, 1, 0)), (True, (1, 1, 1))],
[(True, (1, 1, 0)), (True, (1, 2, 0)), (True, (1, 2, 1))]], [[(True, (1, 0, 0)),
(True, (2, 0, 0)), (True, (2, 0, 1))], [(True, (2, 0, 0)), (True, (2, 1, 0)),
(True, (2, 1, 1))], [(True, (2, 1, 0)), (True, (2, 2, 0)), (True, (2, 2, 1))]]]
    -> Pass
    Expected: [[[True, False, False], [False, False, False], [False, False,
False]], [[False, False, False], [False, False, False], [False, False, False]],
[[False, False, False], [False, False, False], [False, False, False]]]
    Result: [[[(True, (-1, -1, -1)), (False, (0, 0, 0)), (False, (0, 0, 0))],
[(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0,
0)), (False, (0, 0, 0)), (False, (0, 0, 0))]], [[(False, (0, 0, 0)), (False, (0,
0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0,
0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))]], [[(False,
(0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False,
(0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False,
(0, 0, 0))]]]
    -> Pass
    Expected: [[[True, False, False], [False, False, False], [False, False,
False]], [[False, False, False], [False, True, False], [False, False, False]],
[[False, False, False], [False, False, False], [False, False, True]]]
    Result: [[[(True, (-1, -1, -1)), (False, (0, 0, 0)), (False, (0, 0, 0))],
[(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0,
0)), (False, (0, 0, 0)), (False, (0, 0, 0))]], [[(False, (0, 0, 0)), (False, (0,
0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (True, (0, 0, 0)), (False, (0,
0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))]], [[(False,
(0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False,
(0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (True,
(1, 1, 1))]]]
    -> Pass
    Expected: [[[True, True, True], [True, True, True], [True, True, True]],
[[False, False, False], [False, False, False], [False, False, False]], [[False,
False, False], [False, False, False], [False, False, False]]]
```

```
    Result: [[[(True, (-1, -1, -1)), (True, (0, 0, 0)), (True, (0, 0, 1))],
[(True, (0, 0, 0)), (True, (0, 1, 0)), (True, (0, 1, 1))], [(True, (0, 1, 0)),
(True, (0, 2, 0)), (True, (0, 2, 1))]], [[(False, (0, 0, 0)), (False, (0, 0, 0)),
(False, (0, 0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0,
0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))]], [[(False,
(0, 0, 0)), (False, (0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False,
(0, 0, 0)), (False, (0, 0, 0))], [(False, (0, 0, 0)), (False, (0, 0, 0)), (False,
(0, 0, 0))]]]
    -> Pass
```

Traceback:

```
----------TraceBack Test----------
Problem 0:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Moves: [(1, 0, 0), (0, 1, 0), (0, 0, 1)]
How to Get to Point (2, 2, 2)
    Move 0: Start -> (0, 0, 0)
    Move 1: (0, 0, 0) -> (1, 0, 0)
    Move 2: (1, 0, 0) -> (2, 0, 0)
    Move 3: (2, 0, 0) -> (2, 1, 0)
    Move 4: (2, 1, 0) -> (2, 2, 0)
    Move 5: (2, 2, 0) -> (2, 2, 1)
    Move 6: (2, 2, 1) -> (2, 2, 2)
Reachable Points from (0, 0, 0):
    (0, 0, 0)
    (0, 0, 1)
    (0, 0, 2)
    (0, 1, 0)
    (0, 1, 1)
    (0, 1, 2)
    (0, 2, 0)
    (0, 2, 1)
    (0, 2, 2)
    (1, 0, 0)
    (1, 0, 1)
    (1, 0, 2)
    (1, 1, 0)
    (1, 1, 1)
    (1, 1, 2)
    (1, 2, 0)
    (1, 2, 1)
    (1, 2, 2)
    (2, 0, 0)
    (2, 0, 1)
    (2, 0, 2)
    (2, 1, 0)
    (2, 1, 1)
    (2, 1, 2)
    (2, 2, 0)
    (2, 2, 1)
    (2, 2, 2)
```

```
Problem 1:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Moves: [(0, 0, 0)]
How to Get to Point (2, 2, 2)
Point Not Reachable
Reachable Points from (0, 0, 0):
    (0, 0, 0)


Problem 2:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Moves: [(1, 1, 1)]
How to Get to Point (2, 2, 2)
    Move 0: Start -> (0, 0, 0)
    Move 1: (0, 0, 0) -> (1, 1, 1)
    Move 2: (1, 1, 1) -> (2, 2, 2)
Reachable Points from (0, 0, 0):
    (0, 0, 0)
    (1, 1, 1)
    (2, 2, 2)


Problem 3:
Grid Size: (3, 3, 3)
Starting Point: (0, 0, 0)
Moves: [(0, 1, 0), (0, 0, 1)]
How to Get to Point (2, 2, 2)
Point Not Reachable
Reachable Points from (0, 0, 0):
    (0, 0, 0)
    (0, 0, 1)
    (0, 0, 2)
    (0, 1, 0)
    (0, 1, 1)
    (0, 1, 2)
    (0, 2, 0)
    (0, 2, 1)
    (0, 2, 2)
```

Double Unbounded Knapsack Problem (3)

Known Problems:

```
Problem 0:
K1, K2: 9, 11,
Items: [3, 2]
Result: True

Problem 1:
K1, K2: 12, 32
```

```
Items: [5, 8]
Result: False

Problem 2:
K1, K2: 1, 23
Items: [1]
Result: True

Problem 3:
K1, K2:  0, 0
Items: [1]
Result: True
```

Traceback Problems:

```
Problem 0:
K1, K2: 9, 11,
Items: [3, 2]

Problem 1:
K1, K2: 120, 123
Items: [10, 9, 4]

Problem 2:
K1, K2: 1, 23
Items: [1]

Problem 3:
K1, K2:  0, 0
Items: [1]
```

Equality Results:

```
----------Equality Test----------
Testing Problem Size 4
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: False -> Pass
```

```
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
Testing Problem Size 8
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: True -> Pass
    Function 1: True -> Pass
    Function 2: True -> Pass
    Function 0: False -> Pass
    Function 1: False -> Pass
    Function 2: False -> Pass
```

DP Results:

```
----------DP Test----------
Testing DP Algorithm:
    Expected: True, Result: True -> Pass
    Expected: False, Result: False -> Pass
    Expected: True, Result: True -> Pass
    Expected: True, Result: True -> Pass
```

Traceback:

```
```

```
----------TraceBack Test----------
Problem 0:
Possible Items:
[3, 2]
Items In Knapsack 1 (9)
[2, 2, 2, 3]
Items in Knapsack 2 (11)
[2, 2, 2, 2, 3]


Problem 1:
Possible Items:
[10, 9, 4]
Items In Knapsack 1 (120)
[4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 10,
10]
Items in Knapsack 2 (123)
[4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 9,
10]


Problem 2:
Possible Items:
[1]
Items In Knapsack 1 (1)
[1]
Items in Knapsack 2 (23)
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]


Problem 3:
Possible Items:
[1]
Items In Knapsack 1 (0)
[]
Items in Knapsack 2 (0)
[]



Process finished with exit code 0
```

2D Grid Problem (4)

Known Problems:

```
Problem 0:
Grid: [[0, 7, 8], [12, 0, 0], [1, 1, 1]]
Result: 20

Problem 1:
Grid: [[12, 12, 12], [1, 7, 9], [52, 0, 0]]
Result: 71
```

```
Problem 2:
Grid: [[8, 5, 6], [6, 5, 8], [12, 7, 3]]
Result: 25

Problem 3:
Grid: [[0, 0, 0], [1, 1, 18], [99, 0, 0]]
Result: 100
```

Traceback Problems:

```
Problem 0:
Grid: [[0, 7, 8], [12, 0, 0], [1, 1, 1]]

Problem 1:
Grid: [[12, 12, 12], [1, 7, 9], [52, 0, 0]]

Problem 2:
Grid: [[8, 5, 6], [6, 5, 8], [12, 7, 3]]

Problem 3:
Grid: [[0, 0, 0], [1, 1, 18], [99, 0, 0]]
```

Equality Results:

```
----------Equality Test----------
Testing Problem Size 4
    Function 0: 62 -> Pass
    Function 1: 62 -> Pass
    Function 2: 62.0 -> Pass
    Function 0: 71 -> Pass
    Function 1: 71 -> Pass
    Function 2: 71.0 -> Pass
Testing Problem Size 8
    Function 0: 176 -> Pass
    Function 1: 176 -> Pass
    Function 2: 176.0 -> Pass
    Function 0: 144 -> Pass
    Function 1: 144 -> Pass
    Function 2: 144.0 -> Pass
Testing Problem Size 16
    Function 0: 306 -> Pass
    Function 1: 306 -> Pass
    Function 2: 306.0 -> Pass
    Function 0: 314 -> Pass
    Function 1: 314 -> Pass
    Function 2: 314.0 -> Pass
```

DP Results:

```
----------DP Test----------
Testing DP Algorithm:
    Expected: 20, Result: 20.0 -> Pass
    Expected: 71, Result: 71.0 -> Pass
    Expected: 25, Result: 25.0 -> Pass
    Expected: 100, Result: 100.0 -> Pass
```

Traceback:

```
----------TraceBack Test----------
Problem 0:
Max Score: 20.0
Starting Board:
================================
-------------------------
|     0 |    12 |     1 |
-------------------------
|     7 |     0 |     1 |
-------------------------
|     8 |     0 |     1 |
-------------------------
Move 0
================================
-------------------------
|     0 |    12 |     1 |
-------------------------
|  X(7) |     0 |     1 |
-------------------------
|     8 |     0 |     1 |
-------------------------
Move 1
================================
-------------------------
|     0 | X(19) |     1 |
-------------------------
|     7 |     0 |     1 |
-------------------------
|     8 |     0 |     1 |
-------------------------
Move 2
================================
-------------------------
|     0 |    12 |     1 |
-------------------------
|     7 |     0 | X(20) |
-------------------------
|     8 |     0 |     1 |
-------------------------
Overall Path:
================================
-------------------------
|     0 | X(19) |     1 |
-------------------------
|  X(7) |     0 | X(20) |
```

```
-------------------------
|     8 |     0 |     1 |
-------------------------


Problem 1:
Max Score: 71.0
Starting Board:
==============================
-------------------------
|    12 |     1 |    52 |
-------------------------
|    12 |     7 |     0 |
-------------------------
|    12 |     9 |     0 |
-------------------------
Move 0
==============================
-------------------------
| X(12) |     1 |    52 |
-------------------------
|    12 |     7 |     0 |
-------------------------
|    12 |     9 |     0 |
-------------------------
Move 1
==============================
-------------------------
|    12 |     1 |    52 |
-------------------------
|    12 | X(19) |     0 |
-------------------------
|    12 |     9 |     0 |
-------------------------
Move 2
==============================
-------------------------
|    12 |     1 | X(71) |
-------------------------
|    12 |     7 |     0 |
-------------------------
|    12 |     9 |     0 |
-------------------------
Overall Path:
==============================
-------------------------
| X(12) |     1 | X(71) |
-------------------------
|    12 | X(19) |     0 |
-------------------------
|    12 |     9 |     0 |
-------------------------


Problem 2:
```

```
Max Score: 25.0
Starting Board:
==============================
-------------------------
|     8 |     6 |    12 |
-------------------------
|     5 |     5 |     7 |
-------------------------
|     6 |     8 |     3 |
-------------------------
Move 0
==============================
-------------------------
|  X(8) |     6 |    12 |
-------------------------
|     5 |     5 |     7 |
-------------------------
|     6 |     8 |     3 |
-------------------------
Move 1
==============================
-------------------------
|     8 |     6 |    12 |
-------------------------
|     5 | X(13) |     7 |
-------------------------
|     6 |     8 |     3 |
-------------------------
Move 2
==============================
-------------------------
|     8 |     6 | X(25) |
-------------------------
|     5 |     5 |     7 |
-------------------------
|     6 |     8 |     3 |
-------------------------
Overall Path:
==============================
-------------------------
|  X(8) |     6 | X(25) |
-------------------------
|     5 | X(13) |     7 |
-------------------------
|     6 |     8 |     3 |
-------------------------


Problem 3:
Max Score: 100.0
Starting Board:
==============================
----------------------------
|      0 |      1 |     99 |
----------------------------
```

```
|      0 |      1 |      0 |
----------------------------
|      0 |     18 |      0 |
----------------------------
Move 0
==============================
----------------------------
|   X(0) |      1 |     99 |
----------------------------
|      0 |      1 |      0 |
----------------------------
|      0 |     18 |      0 |
----------------------------
Move 1
==============================
----------------------------
|      0 |      1 |     99 |
----------------------------
|      0 |   X(1) |      0 |
----------------------------
|      0 |     18 |      0 |
----------------------------
Move 2
==============================
----------------------------
|      0 |      1 | X(100) |
----------------------------
|      0 |      1 |      0 |
----------------------------
|      0 |     18 |      0 |
----------------------------
Overall Path:
==============================
----------------------------
|   X(0) |      1 | X(100) |
----------------------------
|      0 |   X(1) |      0 |
----------------------------
|      0 |     18 |      0 |
----------------------------
```

Nim Problem (5)

Known Problems:

```
Problem 0:
Stones: [2, 12, 7, 14]
Result: 26

Problem 1:
Stones: [32, 8, 27, 54, 2, 8, 17, 10]
Result: 104

Problem 2:
Stones: [232, 123, 100, 305, 412, 121, 90]
```

```
Rewsult: 758

Problem 3:
Stones: [0, 0, 0, 0, 1, 0, 0, 0, 0]
Result: 0
```

Traceback Problems:

```
Problem 0:
Stones: [2, 12, 7, 14]

Problem 1:
Stones: [32, 8, 27, 54, 2, 8, 17, 10]

Problem 2:
Stones: [232, 123, 100, 305, 412, 121, 90]

Problem 3:
Stones: [0, 0, 0, 0, 1, 0, 0, 0, 0]
```

Equality Results:

```
----------Equality Test----------
Testing Problem Size 2
    Function 0: 94 -> Pass
    Function 1: 94 -> Pass
    Function 2: 94.0 -> Pass
Testing Problem Size 4
    Function 0: 135 -> Pass
    Function 1: 135 -> Pass
    Function 2: 135.0 -> Pass
Testing Problem Size 8
    Function 0: 207 -> Pass
    Function 1: 207 -> Pass
    Function 2: 207.0 -> Pass
Testing Problem Size 16
    Function 0: 424 -> Pass
    Function 1: 424.0 -> Pass
    Function 2: 424.0 -> Pass
```

DP Results:

```
----------DP Test----------
Testing DP Algorithm:
    Expected: 26, Result: 26.0 -> Pass
    Expected: 104, Result: 104.0 -> Pass
    Expected: 758, Result: 758.0 -> Pass
    Expected: 0, Result: 0.0 -> Pass
```

Traceback:

```
----------TraceBack Test----------
Problem 0:
```

```
GAME START:
[2, 12, 7, 14]
Player 1: 14

[2, 12, 7]
Player 2: 2

[12, 7]
Player 1: 12

[7]
Player 2: 7

[]


Problem 1:
GAME START:
[32, 8, 27, 54, 2, 8, 17, 10]
Player 1: 32

[8, 27, 54, 2, 8, 17, 10]
Player 2: 8

[27, 54, 2, 8, 17, 10]
Player 1: 10

[27, 54, 2, 8, 17]
Player 2: 27

[54, 2, 8, 17]
Player 1: 54

[2, 8, 17]
Player 2: 17

[2, 8]
Player 1: 8

[2]
Player 2: 2

[]


Problem 2:
GAME START:
[232, 123, 100, 305, 412, 121, 90]
Player 1: 232

[123, 100, 305, 412, 121, 90]
Player 2: 123

[100, 305, 412, 121, 90]
Player 1: 100
```

```
[305, 412, 121, 90]
Player 2: 90

[305, 412, 121]
Player 1: 305

[412, 121]
Player 2: 412

[121]
Player 1: 121

[]


Problem 3:
GAME START:
[0, 0, 0, 0, 1, 0, 0, 0, 0]
Player 1: 0

[0, 0, 0, 1, 0, 0, 0, 0]
Player 2: 0

[0, 0, 1, 0, 0, 0, 0]
Player 1: 0

[0, 1, 0, 0, 0, 0]
Player 2: 0

[0, 1, 0, 0, 0]
Player 1: 0

[1, 0, 0, 0]
Player 2: 1

[0, 0, 0]
Player 1: 0

[0, 0]
Player 2: 0

[0]
Player 1: 0

[]
```