Algorithm Description

<u>Read & Clean Data</u>

The first stage in the algorithm involves reading the textural description of each Raven's Progressive Matrix problem into memory so it could be worked with in Python, and cleaning extraneous characters from the data. Since the raw data consisted of .txt files, '\n' characters (representing new lines), and '\t' characters (representing tabs) has to be removed.

<u>Parse Data & Populate Dataframe</u>

Once cleaned, the data was parsed into several sparse feature vectors, which were ultimately written to a Pandas dataframe. The CASE and SHAPE LABEL features in the below dataframe examples were hardcoded – the data structure of these features remains the same throughout every file, with the exception of problem #15, where there are 5 shapes instead of 3. This was caught after the algorithm was already written. For the remaining features, dynamic code was written to fully populate the vectors with values and nulls, based on the presence or absence of certain attributes in the files. The dimensions of these vectors were kept consistent by replacing missing values in the text files with null characters ('-').

In the sparse matrices below, each row represents a shape in a given image ('case'). For example, a row might represent all the metadata associated with the Z shape in the A case, or the Y shape in the 3rd case (answer case), etc. Each column represents a feature. The CASE column holds the case labels (A, B, C, 1, 2, 3, 4, 5, 6), the SHAPE LABEL column holds the values Z, Y, or X, which denote different shapes in a given case, and the remaining columns represent attribute metadata for those shapes (e.g. SHAPE can be circle, square, triangle, etc.) The presence of null values '-', representing an absence of data for a particular feature, made the generate and test technique for solving the problem more straightforward and easier to implement.

Problem #01 Sparse Matrix

	CASE	SHAPE LABEL	SHAPE	SIZE	FILL	INSIDE	ABOVE	OVERLAPS	ANGLE	LEFT-OF	VERTICAL-FLIP
0	A	Z	circle	-	no	-	-	-	-	-	-
1	A	-	-	-	-	-	-	-	-	-	-
2	A	-	-	-	-	-	-	-	-	-	-
3	B	Z	circle	-	yes	-	-	-	-	-	-
4	B	-	-	-	-	-	-	-	-	-	-
5	B	-	-	-	-	-	-	-	-	-	-
6	C	Z	square	-	no	-	-	-	-	-	-
7	C	-	-	-	-	-	-	-	-	-	-
8	C	-	-	-	-	-	-	-	-	-	-
9	1	Z	circle	-	yes	-	-	-	-	-	-
10	1	-	-	-	-	-	-	-	-	-	-
11	1	-	-	-	-	-	-	-	-	-	-
12	2	Z	triangle	-	no	-	-	-	-	-	-
13	2	-	-	-	-	-	-	-	-	-	-
14	2	-	-	-	-	-	-	-	-	-	-
15	3	Z	square	-	no	-	-	-	-	-	-
16	3	-	-	-	-	-	-	-	-	-	-
17	3	-	-	-	-	-	-	-	-	-	-
18	4	Z	triangle	-	yes	-	-	-	-	-	-
19	4	-	-	-	-	-	-	-	-	-	-
20	4	-	-	-	-	-	-	-	-	-	-
21	5	Z	square	-	yes	-	-	-	-	-	-
22	5	-	-	-	-	-	-	-	-	-	-
23	5	-	-	-	-	-	-	-	-	-	-
24	6	Z	circle	-	no	-	-	-	-	-	-
25	6	-	-	-	-	-	-	-	-	-	-
26	6	-	-	-	-	-	-	-	-	-	-

Problem #20 Sparse Matrix

	CASE	SHAPE LABEL	SHAPE	SIZE	FILL	INSIDE	ABOVE	OVERLAPS	ANGLE	LEFT-OF	VERTICAL-FLIP
0	A	Z	circle	large	no	-	-	-	0	-	-
1	A	Y	square	medium	no	Z	-	-	0	-	-
2	A	X	triangle	small	no	Z,Y	-	-	0	-	-
3	B	Z	circle	large	yes	-	-	-	0	-	-
4	B	Y	square	medium	no	Z	-	-	0	-	-
5	B	X	triangle	small	no	Z,Y	-	-	180	-	-
6	C	Z	square	large	no	-	-	-	0	-	-
7	C	Y	circle	medium	no	Z	-	-	0	-	-
8	C	X	Pac-Man	small	no	Z,Y	-	-	270	-	-
9	1	Z	square	large	yes	-	-	-	0	-	-
10	1	Y	circle	medium	no	Z	-	-	0	-	-
11	1	X	Pac-Man	small	no	Z,Y	-	-	270	-	-
12	2	Z	square	large	yes	-	-	-	0	-	-
13	2	Y	circle	medium	no	Z	-	-	0	-	-
14	2	X	Pac-Man	small	yes	Z,Y	-	-	90	-	-
15	3	Z	square	large	no	-	-	-	0	-	-
16	3	Y	circle	medium	yes	Z	-	-	0	-	-
17	3	X	Pac-Man	small	no	Z,Y	-	-	90	-	-
18	4	Z	square	large	no	-	-	-	0	-	-
19	4	Y	circle	medium	no	Z	-	-	0	-	-
20	4	X	Pac-Man	small	no	Z,Y	-	-	270	-	-
21	5	Z	square	large	yes	-	-	-	0	-	-
22	5	Y	circle	medium	no	Z	-	-	0	-	-
23	5	X	Pac-Man	small	no	Z,Y	-	-	90	-	-
24	6	Z	square	large	yes	-	-	-	0	-	-
25	6	Y	circle	medium	no	Z	-	-	0	-	-
26	6	X	Pac-Man	small	yes	Z,Y	-	-	270	-	-

Semantic Network, Generate & Test

To solve for the fourth image ('image D') in each text file, semantic networks representing the relationships between images were generated and tested. First, a semantic network representing the relationship between case A and case B was generated. This knowledge representation serves as the 'testing' data for the algorithm. Each A row in the dataframe is compared to each B row. The relationships between overall rows are first enumerated, as so: e.g. $A_Z \rightarrow B_Z$ = 'changed', $A_Y \rightarrow B_Y$ = 'unchanged', etc. The subscripts here represent shapes in each image. If the entire row associated with an A shape is equivalent to the entire row associated with a B shape, there is no need to compare individual feature values, and the value 'ROW UNCHANGED' is stored. If, on the other hand, there are some changes between the two rows, then each individual feature value is compared and enumerated. In this context, 'enumerated' means qualifying the relationships between the two individual feature values as 'changed', 'unchanged', 'added', or 'removed'. For example, if $A_{Z, \text{Shape}}$ = 'circle' and $B_{Z, \text{Shape}}$ = 'square' $\rightarrow$ relationship = 'changed'. If $A_{Z, \text{Inside}}$ = 'Y' and $B_{Z, \text{Inside}}$ = '-' $\rightarrow$ relationship = 'removed'. An example data structure representing the relationship between two CASES (e.g. A and B) might look like the following: [['changed', 'removed', 'removed', 'added', 'unchanged', 'removed', 'unchanged', 'unchanged', 'changed', 'unchanged', 'unchanged'], 'ROW UNCHANGED', 'ROW UNCHANGED']]. This indicates that there are some changes to the Z shape metadata between the two cases being compared, but all the metadata associated with the Y and X shapes are exactly the same between cases. The first nested list in this structure describes the actual per-value changes.

After this knowledge representation describing the relationships between cases A and B is generated for each file, it can be tested against the same relationships that exist between case C, and every possible answer. The same data structure described above is generated for the following case pairs: C1, C2, C3, C4, C5, and C6. Then these data structures are compared to that of the AB pair. If the relationships match exactly, the algorithm predicts that case as the answer. For example, if AB == C3, then case 3 is likely the correct answer. **Note that this is a naïve algorithm.** It simply looks for value changes at a high level, and does not quantify the change itself in detail. For example, if the ANGLE attribute for a shape changed from 90 to 180 from one case to another, the change is simply represented as 'changed', rather than 'changed by 90 degrees'. For this reason, **the algorithm only successfully solves 10 of 20 problems.** It is possible that the algorithm may not find an answer for a given file, if there is no exact match between case pairs. It is also possible that it predicts the incorrect answer if the problem requires a **more granular** knowledge representation than that provided by the simple descriptors 'changed', 'unchanged', 'added', and 'removed'. This is the case in problems where the knowledge representation of the AB pair matches **more than one** of the potential answer pairs (e.g. AB == C1 AND AB == C3). In these cases, the algorithm may predict incorrectly. With this said, the naïve approach does well to solve roughly half of the problems, and those that it fails to solve can be examined separately using more granular knowledge representations.

Results

```
PROBLEM #01
ANSWER: 5
PREDICTED ANSWER: 5
--------------------------------------------------------------------------

PROBLEM #02
ANSWER: 6
PREDICTED ANSWER: 6
--------------------------------------------------------------------------

PROBLEM #03
ANSWER: 4
PREDICTED ANSWER: 4
--------------------------------------------------------------------------

PROBLEM #04
ANSWER: 3
PREDICTED ANSWER: 3
--------------------------------------------------------------------------

PROBLEM #05
ANSWER: 2
PREDICTED ANSWER: 2
--------------------------------------------------------------------------

PROBLEM #06
ANSWER: 5
PREDICTED ANSWER: 5
--------------------------------------------------------------------------

PROBLEM #07
ANSWER: 2
PREDICTED ANSWER: 2
--------------------------------------------------------------------------

PROBLEM #08
ANSWER: 1
No answer found
--------------------------------------------------------------------------

PROBLEM #09
ANSWER: 1
PREDICTED ANSWER: 1
--------------------------------------------------------------------------

PROBLEM #10
ANSWER: 4
PREDICTED ANSWER: 2
--------------------------------------------------------------------------
```

PROBLEM #11
ANSWER: 4
PREDICTED ANSWER: 1

PROBLEM #12
ANSWER: 6
PREDICTED ANSWER: 3

PROBLEM #13
ANSWER: 2
PREDICTED ANSWER: 1

PROBLEM #14
ANSWER: 3
No answer found

PROBLEM #15
ANSWER: 6
PREDICTED ANSWER: 1

PROBLEM #16
ANSWER: 3
No answer found

PROBLEM #17
ANSWER: 1
PREDICTED ANSWER: 1

PROBLEM #18
ANSWER: 2
PREDICTED ANSWER: 4

PROBLEM #19
ANSWER: 1
No answer found

PROBLEM #20
ANSWER: 5
PREDICTED ANSWER: 5
