## **Documentation:**

The Program has 3 main Classes: Grid, State, Rule . It simulates Childrens' Puzzle Book.

Main points in the Code:

Main function: create a Grid, an array of words (from the Input) and an Initiate state and call flailWildy function.

<u>flailWildy</u>: (in the first version – wordSearchMaker.py) main while loop which checks each time if the game was over and if not, locates in the grid the next word from the "bank".

<u>backtrack</u>: (in the second version – wordSearchMaker2.py) this function consistently checks if the goal is achieved, or any possible fail that prevents us continue keep searching for a solution in a specific branch. Otherwise, continues do deeper at the search tree and update 3 global variables each iteration.

<u>allCandidates</u>: there are 8 possible directions for each word and MxN initiate places on the grid. So, it calls precondition function 8MN times and create a list with possible rules for apply.

<u>precondition</u>: A Boolean type function which checks:

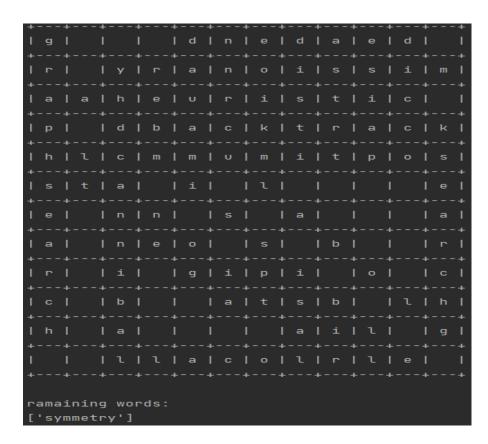
- 1. if writing the word on the grid is "out of bounds"
- 2. if writing the word on the grid is "overlap another word"

<u>applyRule</u>: locates the word on the grid by the following steps:

- 1. Creating a new updates list of words
- 2. deep copy of the previous grid
- 3. placing the current word in the grid
- 4. Creating a new State and return it.

## Outputs for wordSearchMaker.py with 12x12 Grid:

+-																								+
I.									m		U		m		i		t		р		0		m	1
+-		+-														+-		+-						+
ı		1	r	١.		1		1		1	ι	1		٠,		1				1	d	١.	i	1
+-																								+
ı	k	1	а	1		1	i	1	ι	1	0	1	У	-		1		1		1	ι	1		П
+-																								+
ı		1	þ	1	а	1	t	1	i	1		1	r	-		1	а	1		1	а	1		1
+-																								+
ı	а	ı	h	1	g	ı		1		1	а	ı	t	-	d	ı	n	ı	h	1	n	1	i	ı
+-																								+
ı	r	ı		ı		ı	i	1	р	1	ι	ı		-		ı	n	ı		1	0	1		I
+-																								+
ı	t	ı		1	n	ı	r	Т		Т	n	ı	m	-		ı	i	ı	r	Т	i	ı	n	I
+-																								+
I	k	ı	а	1	t	ı	U	ı	d	Т		ı	m	-		ı	b	ı	а	Т	t	ı	а	I
+-																								+
ı		ı	r	Т		ı		Т		Т		ı	У	-1		ı	а	Т		Т	а	Т	r	I
+-																								+
ı	а			Т			h	Т		Т				П			ι			Т	r		У	I
+-																								+
I	b	ı	h	ı	а	ı	d	1	m	1	i	ı		-		ı	i	ı	b	1	ι	ı		I
+-																								+
I								ı							ι		0		b		а		ι	1
+-																								+
		air	nir	ıg	wc	oro	s:																	



## Outputs for wordSearchMaker2.py with 12x12 Grid:

```
Number of Failures:
0
Number of backtrack calls:
16
```

## Outputs for wordSearchMaker2.py with 11x11 Grid:

```
placed the word: "admissible" at: (0, 0), direction: (0, 1)
placed the word: "agent" at: (0, 0), direction: (1, 0)
placed the word: "backtrack" at: (0, 5), direction: (0, 1)
placed the word: "cannibal" at: (0, 6), direction: (0, 1)
placed the word: "deadend" at: (0, 7), direction: (0, 1)
placed the word: "global" at: (0, 1), direction: (0, 1)
placed the word: "graphsearch" at: (0, 8), direction: (0, 1)
placed the word: "heuristic" at: (0, 9), direction: (0, 1)
placed the word: "lisp" at: (0, 10), direction: (0, 1)
placed the word: "local" at: (1, 2), direction: (0, 1)
placed the word: "missionary" at: (10, 3), direction: (0, -1)
placed the word: "rational" at: (1, 4), direction: (0, 1)
placed the word: "search" at: (5, 8), direction: (0, 1)
placed the word: "symmetry" at: (10, 0), direction: (1, 0)
```

```
Number of Failures:
106
Number of backtrack calls:
122
```