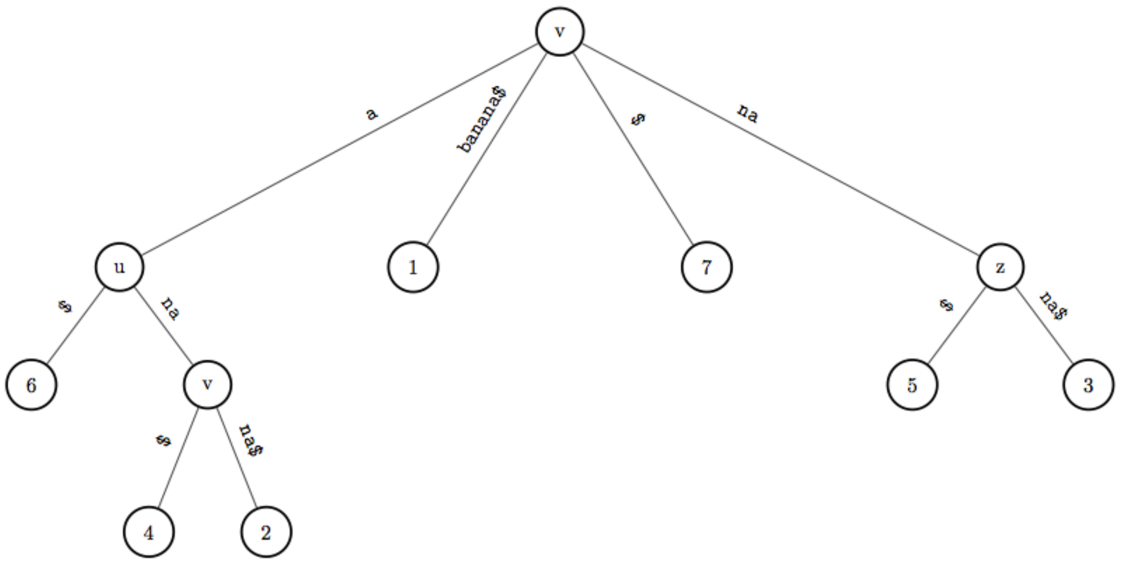
TranslateToLatexTree

This is a python script file that translates tree-graph information stored in a .txt file to complicated LaTeX code, which can be compiled into a pretty tree graph in LaTeX editor (ex. Overleaf).

Basic usage

If you want to draw this tree:



Then you should create a .txt file where each line describes a level of the tree, starting from the root:

```
line 1: v
```

In the second line you want to describe 4 nodes. Each node description is separated by the others with the symbol '|' and contains the label of the incoming edge (if any) and the label of the node, encoded with this syntax:

```
[label of the incoming edge] ; [label of the node]
```

Then the second will be:

```
line 2: a;u | banana$;1 | $;7 | na;z
```

In the third line, you have nodes with different parents: the description you will put at the beginning of the line will refer to the leftmost node at the upper level (in this case *u*). <u>You can change the parent node using the symbol '@': the parent will become the next parent node on the right</u> (in this case *1*). If the next parent node does not have child nodes, leave the description empty.

Third line will be:

```
line 3: $;6 | na;v @  @  @ $;5 | na$;3
```

Same for the fourth level/line, you have to specify the parent for each description, considering the node that you
created in the previous level: the previous level contains four nodes and only the second has child, then:

```
line 4: @ $;4 | na$;2 @ @
```

Special Syntax for Discrete Mathematical Formula

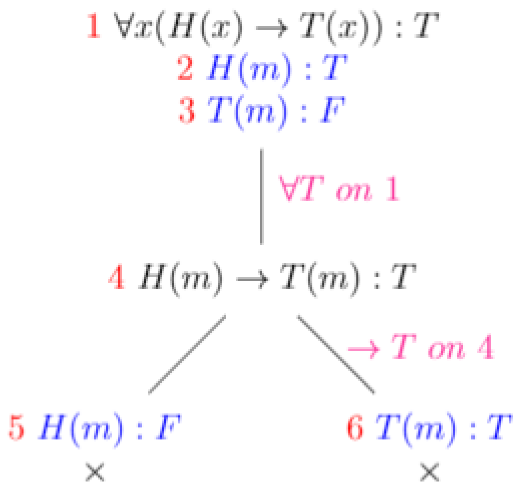
I want to generate some Formula of Discrete Mathematic. There are a lot of special sign here. Hence I defined some Syntax especially:

```
        dic = {"toRight": "\\rightarrow",
               "toLeft": "\\leftarrow",
               "double": "\\leftrightarrow",
               "or": "\\vee",
               "and": "\\wedge ",
               "not": "\\neg",
               "all": "\\forall",
               "exists": "\\exists",
               "cross": "\\times",
               "circle": "\\bigcirc",
               "sblue": "\\textcolor{blue}{", "eblue": "}",
               "sred": "\\textcolor{red}{", "ered": "}",
               "spink": "\\textcolor{magenta}{", "epink": "}",
               "space": "\\ ",
               "nextLine": "$\\\\$",
               "$$": " "}
```

An example of tree-graph information written in a .txt file:

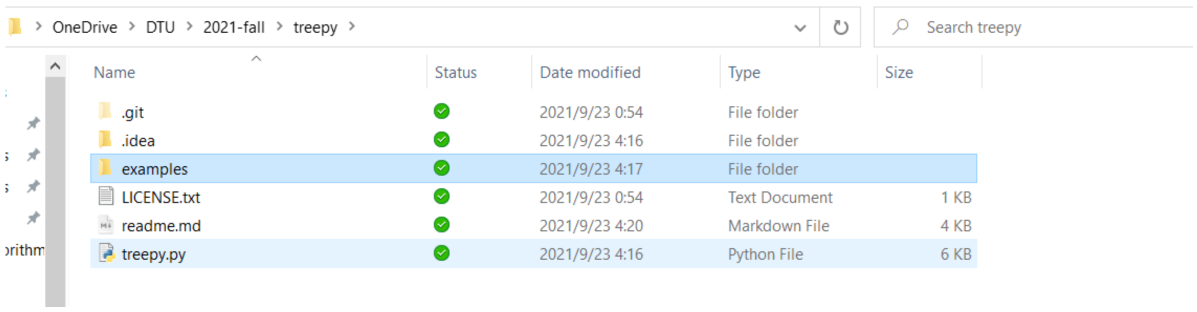
```
sred 1 ered space all x (H(x) toRight T(x)):T nextLine sred 2 ered space sblue
H(m):T eblue nextLine sred 3 ered space sblue T(m):F eblue
all T space on space 1 ; sred 4 ered space H(m) toRight T(m):T
; sred 5 ered space sblue H(m):F eblue nextLine cross | toRight T space on space
4 ; sred 6 ered space sblue T(m):T eblue nextLine cross
```

The generated graph:



How to run this script

My folder looks like this:



OneDrive > DTU > 2021-fall > treepy >

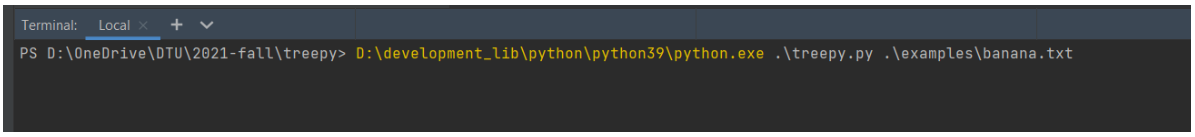
Name	Status	Date modified	Type	Size
.git	✓	2021/9/23 0:54	File folder	
.idea	✓	2021/9/23 4:16	File folder	
examples	✓	2021/9/23 4:17	File folder	
LICENSE.txt	✓	2021/9/23 0:54	Text Document	1 KB
readme.md	✓	2021/9/23 4:20	Markdown File	4 KB
treepy.py	✓	2021/9/23 4:16	Python File	6 KB

In folder `example`, there is a `.txt` file which is used to describe a tree. Its content is mentioned above:

```
sred 1 ered space all x (H(x) toRight T(x)):T nextLine sred 2 ered space sblue
H(m):T eblue nextLine sred 3 ered space sblue T(m):F eblue
all T space on space 1 ; sred 4 ered space H(m) toRight T(m):T
; sred 5 ered space sblue H(m):F eblue nextLine cross | toRight T space on space
4 ; sred 6 ered space sblue T(m):T eblue nextLine cross
```

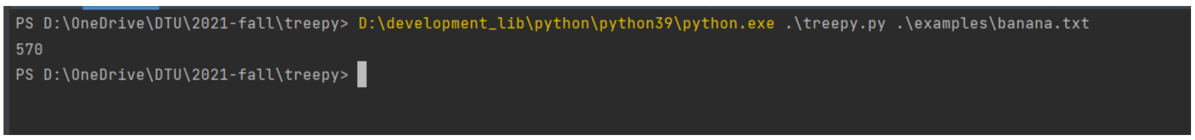
Then I would try the command following:

```
{Python_path} .\treepy.py {tree_description_file}
```



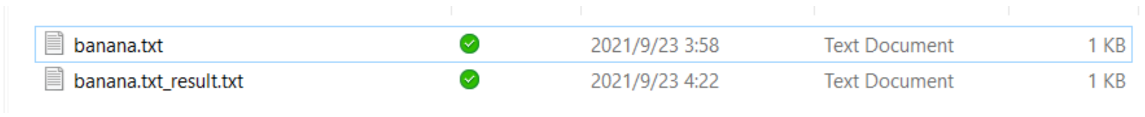
```
PS D:\OneDrive\DTU\2021-fall\treepy> D:\development_lib\python\python39\python.exe .\treepy.py .\examples\banana.txt
```

Run it, and you see:



```
PS D:\OneDrive\DTU\2021-fall\treepy> D:\development_lib\python\python39\python.exe .\treepy.py .\examples\banana.txt
570
PS D:\OneDrive\DTU\2021-fall\treepy>
```

A `xxx_result.txt` file has been generated:



banana.txt	✓	2021/9/23 3:58	Text Document	1 KB
banana.txt_result.txt	✓	2021/9/23 4:22	Text Document	1 KB

The generated Latex code in `xxx_result.txt` looks like:

```
\node[punkt] {$\textcolor{red}{ 1 } \  \forall x (H(x) \rightarrow T(x)):T $\\$
\textcolor{red}{ 2 } \  \textcolor{blue}{ H(m):T } $\\$ \textcolor{red}{ 3 } \
 \textcolor{blue}{ T(m):F }$}
    child {node[punkt] {$\textcolor{red}{ 4 } \  H(m) \rightarrow T(m):T$}
    child {node[punkt] {$\textcolor{red}{ 5 } \  \textcolor{blue}{ H(m):F } $\\$
\times$}
edge from parent
 node[kant,right,pos=.4]{$\textcolor{magenta}{}$}}
    child {node[punkt] {$\textcolor{red}{ 6 } \  \textcolor{blue}{ T(m):T } $\\$
\times$}
edge from parent
 node[kant,right,pos=.4]{$\textcolor{magenta}{\rightarrow T \  on \  4}$}}
edge from parent
 node[kant,right,pos=.4]{$\textcolor{magenta}{\forall T \  on \  1}$}};
```

Copy it and paste it to your overleaf in the following way:

```
\begin{center}
\begin{tikzpicture}[
grow=down,
level 1/.style={sibling distance=5cm,level distance=2.5cm},
level 2/.style={sibling distance=4cm, level distance=2cm},
level 3/.style={sibling distance=4cm, level distance=2cm},
kant/.style={ text centered},
every node/.style={text ragged, inner sep=2mm,align=center},
punkt/.style={ shade, top color=white,
bottom color=white, draw=white, very thick }
]

% Insert generated Latex code here%

\end{tikzpicture}
\end{center}

%end here
```

The tree is drawn using the tikz package. You will have to import the package *tikz*. As you can see the preamble of tikzpicture allows you to personalize your tree .

> @ Author: WenjieDTU & JiananAlvin
>
> Ref: https://github.com/diegoceccarelli/treepy