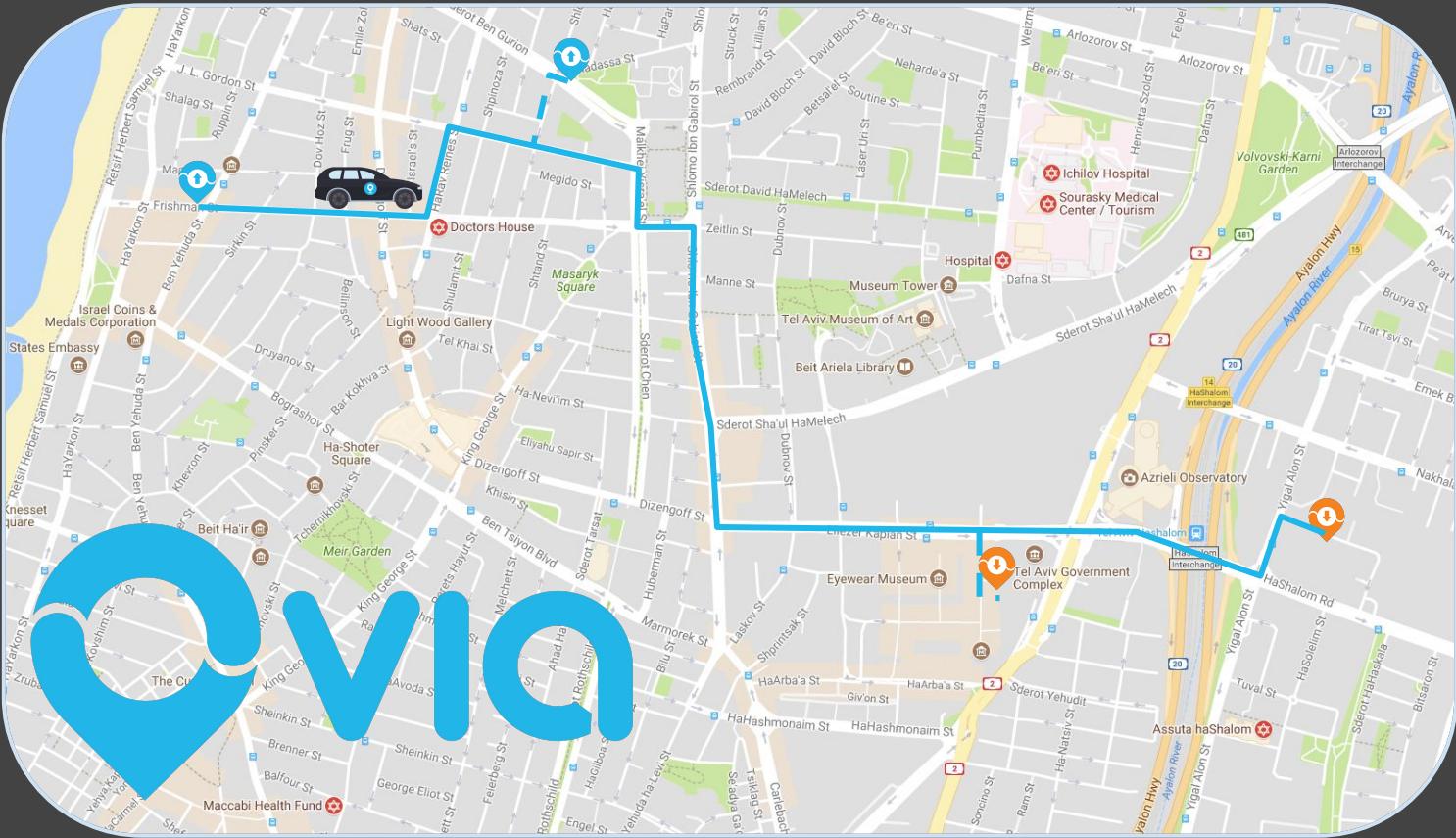


Help your colleagues help themselves - a Sphinx tutorial

Dalya Gartzman

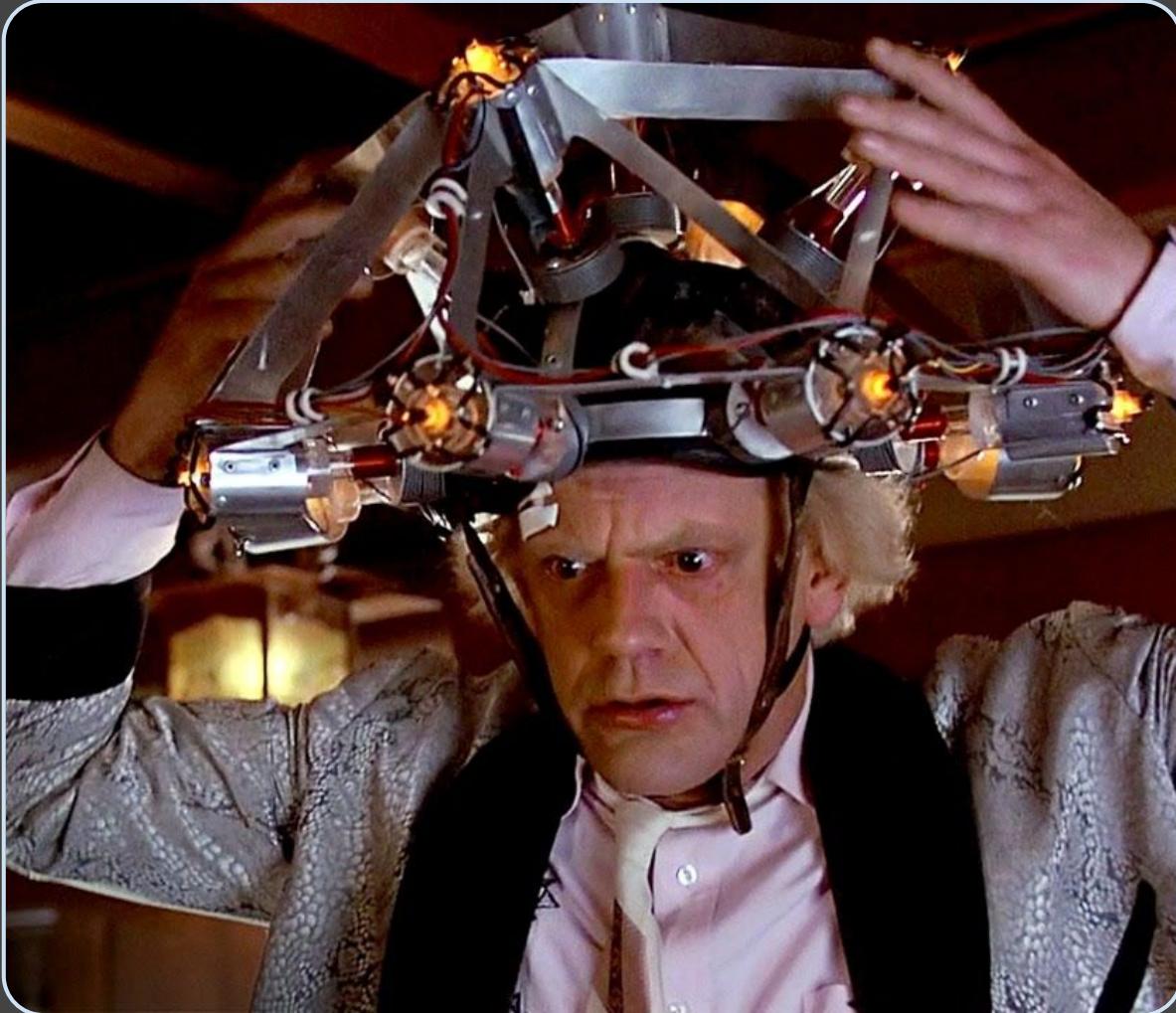
github.com/DalyaG/Sphinx185







L
HTOA - S
CRIA - C
MTIA - X1
MT2A - X2
CRIA - BI
CPIA - BO
CPJA - PU
TYIA - TY
CRIA - SI
LPIA - LC
MT2A - GO



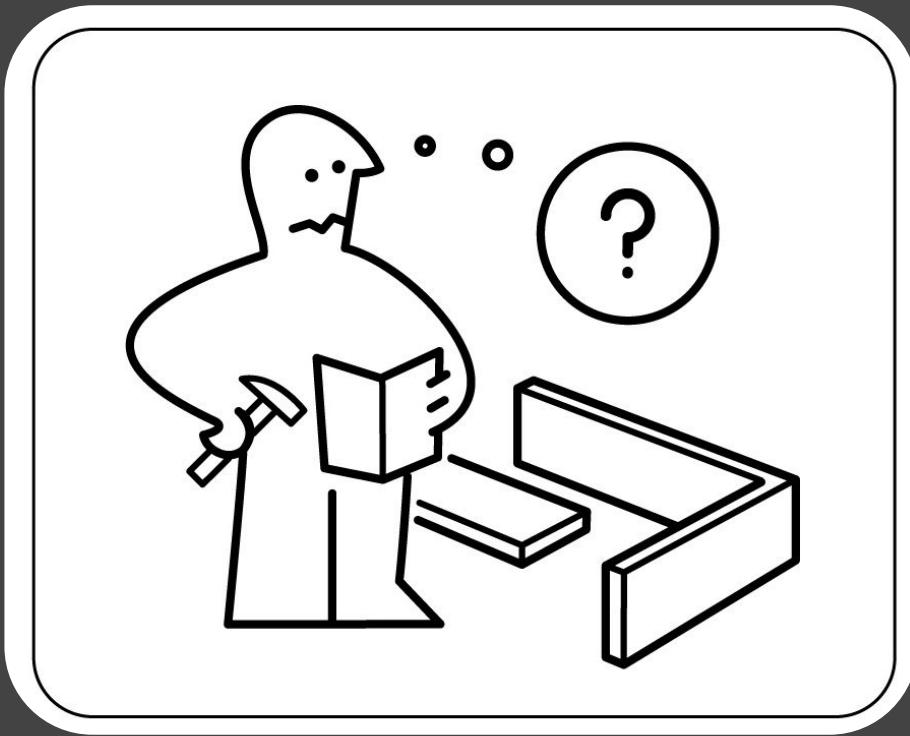




PUBLISH

PUBLISH EVERYTHING





YOU HAD ONE JOB





NEWBIES

A circular sign with a thick red border and a diagonal red slash across it, indicating prohibition. The word "NEWBIES" is written in bold, black, uppercase letters in the center of the white area.



READY PLAYER ONE

Press Start

READY PLAYER ONE

Press Start



Intro - What will we accomplish today?

Intro - What will we accomplish today?

Sphinx185 documentation » [next](#) | [modules](#) | [index](#)



Sphinx185

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [How To Use This for YOUR Next Project :\)](#)

Quick search [Go](#)

Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this project is to help make [Sphinx](#) accessible and easy to use.
To use this for YOUR next project, browse this documentation and follow the instructions [here](#).
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
[Hope you find this useful!](#)
[DalyaG](#)

OK Let's get to business:

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [How To Use This for YOUR Next Project :\)](#)
- [Index](#)

Part I - Overview

Intro - What will we accomplish today?

Sphinx185 documentation »



Sphinx185

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: conf.py

Quick search

Go

The main configurations file: conf.py

Below is the code for `conf.py`, the main configuration file that is used for generating this documentation.

This file was originally made by running `sphinx-quickstart`.

It was modified and customized such that, hopefully, it has a clearer structure, and is easier to re-modify for the sake

```
"""
This is the main file in which the configuration for the documentation is made.

"""
# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.
#
# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'DalyaG'

# -- A few basic configurations

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentat
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']
```

Part II - Main configuration file - conf.py

Intro - What will we accomplish today?

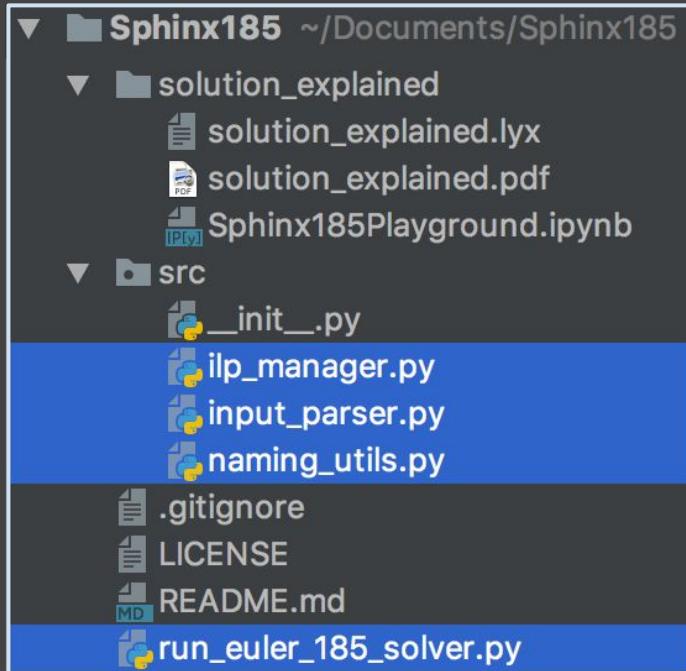
How To Use This for YOUR Next Project :)

1. `pip install sphinx`
2. Copy the `documentation_template_for_your_next_project` folder in this repository, make it a subdirectory in your local project and rename it `documentation`.
3. Edit `conf.py` by searching the pattern `#CHNAGEME#` inside the file and follow the instructions.
4. Edit `index.rst` by following the inline instructions.
5. To add a page to your documentation:
 - a. Create a `.rst` file for the function/module you wish to document (you can use the templates supplied here), and
 - b. Add the name of the `.rst` file to the `toctree` in `index.rst`.
6. In terminal, inside the `documentation` folder, run `make html`.
(TIP OF THE WEEK: actually, always run `make clean html` to clear sphinx cache and build from scratch)
7. To view locally - open the file `documentation/_build/html/index.html` in your browser, and enjoy the read :)
8. To share - you can use [GitHub Pages](#) to host your documentation:
 - a. Copy the content of `documentation/_build/html/` into a new `docs` folder, under the root of the project.
 - b. Create an empty file `.nojekyll` inside `docs` folder
(this tells GitHub Pages to bypass the default `jekyll` themes and use the `html` and `css` in your project)
 - c. Push your changes to master branch.
 - d. In your repository on GitHub, go to “Settings” -> “GitHub Pages” -> “Source” and select “master branch /docs folder”.
 - e. Share your beautiful documentation site at https://<your_git_usrname>.github.io/<project_name>/

Part III - Document YOUR next project

Part I - Overview

Part I - Overview



Part I - Overview

Sphinx185 documentation >



Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this piece is to help make [Sphinx](#) accessible and easy to use.
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
Hope you find this useful!
[@DalyaG](#)

OK Let's get to business:

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [Index](#)

dalyag.github.io/Sphinx185

Part I - Overview

Sphinx185 documentation >

Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this piece is to help make [Sphinx](#) accessible and easy to use.
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
Hope you find this useful!
@DalyaG

OK Let's get to business:

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [Index](#)

index.rst -
the content
of the home
page

Part I - Overview

conf.py -
settings for
the entire
project

Sphinx185 documentation >

Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).
The goal of this piece is to help make [Sphinx](#) accessible and easy to use.
Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).
Hope you find this useful!
@DalyaG

OK Let's get to business:

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)
- [Index](#)

Part I - Overview

Euler's 185th Riddle

In case you are interested, the riddle goes something like this:

Let s^* be a sequence of m **unknown** digits (colors from 0 to 9).

We are given n sequences of m **known** colors, $x_{i,j}$ marks the color of the j^{th} index in sequence i .

For each sequence we are given the number of **correct colors**, that is, indices in the sequence for which $x_{i,j} = s_j^*$.

Our goal is to discover the colors of s^* , which we are promised are unique, given the input.

Part I - Overview

Euler's 185th Riddle

In case you are interested, the riddle goes something like this:

Let s^* be a sequence of m **unknown** digits (colors from 0 to 9).

We are given n sequences of m **known** colors, $x_{i,j}$ marks the i -th

For each sequence we are given the number of **correct colors**, t_i .

Our goal is to discover the colors of s^* , which we are promised a

Solver Runner

`run_euler_185_solver.main()` [[SOURCE](#)]

Run this code to get the solution to 185th problem in Project Euler.

Terminal options:

`-m <sequence_length>, --sequence_length <sequence_length>`

Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

Part I - Overview

Euler's 185th Riddle

In case you are interested:

Let s^* be a sequence.

We are given m .

For each sequence s ,

Our goal is to determine if s is a solution to the riddle.

Solver Runner

Source code for run_euler_185_solver

```
from optparse import OptionParser

from src.ilp_manager import ILPManager
from src.input_parser import input_parser

[docs]def main():
    """
    Run this code to get the solution to 185th problem in Project Euler.

    Terminal options:
    .. option:: -m <sequence_length>, --sequence_length <sequence_length>
        Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

    """
    parser = OptionParser()
    parser.add_option("-m", "--sequence_length",
                      help="Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.")
    options, _ = parser.parse_args()

    m = int(options.sequence_length)
    sequences_list, n_correct_vertices_list = input_parser(m)

    ilp_manager = ILPManager(m)
    ilp_solver, s_star_to_color_edges = ilp_manager.build_ilp_solver(sequences_list, n_correct_vertices_list)
    s_star = ilp_manager.solve_ilp(ilp_solver, s_star_to_color_edges)

    print "Solution is: {}".format('.join([str(i) for i in s_star]))

if __name__ == '__main__':
    main()
```

[solver.main\(\) \[SOURCE\]](#)

Get the solution to 185th problem in Project Euler.

`-m <sequence_length>, --sequence_length <sequence_length>`
Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

Part I - Overview

Euler's 185th Riddle

In case you are interested:

Let s^* be a sequence.

We are given 10 sequences.

For each sequence we are given a color.

Our goal is to find s^* .

Source code for run_euler_185_solver

```
from optparse import OptionParser
from src.ilp_manager import ILPManager
from src.input_parser import input_parser

[docs]def main():
    """
    Run this code to get the solution to 185th problem in Project Euler.

    Terminal options:
    .. option:: -m <sequence_length>, --sequence_length <sequence_length>
        Length of sequence in the riddle. Assume file 'data/input_m.txt' exists.

    ...
    parser = OptionParser()
    parser.add_option("-m", "--sequence_length",
                      help="Length of sequence in the riddle. Assume file 'data/input_m.txt' exists")
    options, _ = parser.parse_args()

    m = int(options.sequence_length)
    sequences_list, n_correct_vertices_list = input_parser(m)

    ilp_manager = ILPManager(m)
    ilp_solver, s_star_to_color_edges = ilp_manager.build_ilp_solver(sequences_list, n_correct_vertices_list)
    s_star = ilp_manager.solve_ilp(ilp_solver, s_star_to_color_edges)

    print "Solution is: {}".format(''.join([str(i) for i in s_star]))

if __name__ == '__main__':
    main()
```

Solver Runner

src.ilp_manager.ILPManager(m) [source]

Model the 185th problem in Project Euler as an ILP (Integer Linear Program)

To instantiate this module, please specify the length on sequences.

build_ilp_solver(sequences_list, n_correct_vertices_list) [source]

Given input sequences, and number or correct vertices in each of them, build an ILP representation of the problem.

Parameters:

- **sequences_list** – List of sequences, each of length self.m, of integers between 0 and 9.
- **n_correct_vertices_list** – Number of correct vertices in each sequence in sequences_list. A vertex is correct if its color is equal to the color of the corresponding vertex in the solution s_star.

Returns:

tuple, containing:

- **ilp_solver**: Pulp instance, holding all the information needed for the solution.
- **s_star_to_color_edges**: The edges (variables) in the ilp_solver.

solve_ilp(ilp_solver, s_star_to_color_edges) [source]

Given a solver with the needed information, solve the ILP and extract the solution to problem 185.

Parameters:

- **ilp_solver** – Pulp instance, holding all the information needed for the solution.
- **s_star_to_color_edges** – The edges (variables) in the ilp_solver.

Returns:

s_star: List of integers that is the solution to problem 185.

Part I - Overview

Euler's 185th Riddle

In case you are interested:

Let s^* be a sequence.

We are given n .

For each sequence s ,

Our goal is to determine if s is a solution of the problem.

Source code for run_euler_185_solver

```
from optparse import OptionParser
from src.ilp_manager import ILPManager
from src.input_parser import input_parser
[docs]def main():
    """
    Run this code in a terminal or Jupyter notebook.
    ... options:
        Length of the sequence.
        ...
    parser = OptionParser()
    parser.add_option("-l", "--length", type="int", dest="length",
                      help="Length of the sequence.", default=10)
    options, sequences = parser.parse_args()
    m = int(options.length)
    sequences_list = sequences.sequences
    ilp_manager = ILPManager(m)
    ilp_solver = ilp_manager.get_ilp_solver()
    s_star = ilp_solver.solve()
    print "Solution found: %s" % s_star
    if __name__ == "__main__":
        main()
```

Solver Runner

[class src.ilp_manager.ILPManager\(m\) \[SOURCE\]](https://dalyag.github.io/Sphinx185/index.html)



Welcome to Sphinx185's documentation!

This documentation follows the solution of the 185th riddle from [Project Euler](#).

The goal of this project is to help make [Sphinx](#) accessible and easy to use.

To use this for YOUR next project, browse this documentation and follow the instructions [here](#).

Feel as free as you can to copy, paste and change anything you see here.

The source code is available [here](#).

Hope you find this useful!

DalyaG

- Euler's 185th Riddle
- Solver Runner
- Input Parser
- Naming Utils
- The ILP Manager class
- The main configurations file: config.py
- How To Use This for YOUR Next Project :)

Part II - Main configuration file - conf.py

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-

# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory --

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project --

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

```
# -- Define and configure non-default extensions ----

# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax ``todo`` some todo command`` in your doctstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
# It will automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to add to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{color}
\definecolor{offwhite}{rgb}{230,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
'''

# -- Options for HTML output --

# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the `graphite` template from here:
# https://github.com/sphinx-doc/sphinx-themes-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
# This theme 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_dalya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In this project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.
```

```
"""
# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build
# sphinx-quickstart on Sat May
#
# DalyaG: This file was heavil
# Hope you find this useful :)
```

```
# -- Define here your working
```

```
import os
import sys
sys.path.append(os.path.abspath(
```

```
# -- Some general info about
```

```
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'
```

```
# -- A few basic configuration
```

```
# The documentation in this pr
# In This project, every auto-
# which is rendered into an
# Get more information here: h
source_suffix = ['.rst']
```

```
# This is the name of the main
# It means that you need to ha
# It will be rendered into an
# (this is a standard name. ch
master_doc = 'index'
```

```
# List of patterns, relative to t
# directories to ignore when looki
# This patterns also effect to html
exclude_patterns = ['_build']
```

```
# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

```
"""
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
```

" in your docstring,
the following:
have an internal link to the source code.
writer highlighting) style.
[getbox.com/pygments.html](#)

the docstrings in your code.
[autodoc.html](#)
docstrings.
order of members as in the code.
of all the class/module members.
directly. It is first rendered to an image.
x document.

[xr/theming.html](#)
added the 'graphite' template from here:
[site.py](#)
ite.css and in html4css1.css,

here,
files,
ject:

```
# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#     and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#   - Crossbox: appears below the TOC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Full details - EXTRA SLIDES

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory ----

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project ----

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations ----

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
# templates_path = ['_templates']
```

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory -----
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project -----
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations -----
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']
```

Templates are html
exmaples for designing
predifined elements

templates_path = ['_templates']

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory -----
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project -----
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations -----
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']
```

Templates are html
examples for designing
predifined elements

We will see:
a template for sidebar

templates_path = ['_templates']

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax "... todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/sphinx-doc/sphinx-themes-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# and deleted graphite.py
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
# you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc',  
'sphinx.ext.imgmath']
```

```
#           It will appear in a highlighted box in the documentation.  
#           In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
#           The source code will have colors defined by the Pygments (syntax highlighting) style.  
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
#           It allows you to automatically generate documentation for the docstrings in your code.  
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{230,238,238}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/sphinx-doc/sphinx/tree/graphite  
# Some adjustments I made to graphite:  
#   - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
#     you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo.)  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
#   - Table of Contents: I chose globaltoc as it is less refined,  
#     and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
#   - Searchbox: appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Let Sphinx know what extensions to use

Part II - Main configuration file - conf.py

'sphinx.ext.autodoc'

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{(230,238,238)}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/rtfd/sphinx-themes/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
# - I copied the graphite.css and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can compare the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

.. autoclass:: src.ilp_manager.ILPManager



class src.ilp_manager.ILPManager(m) [source]

Model the 185th problem in Project Euler as an ILP (Integer Linear Program)

To instantiate this module, please specify the length on sequences.

build_ilp_solver(sequences_list, n_correct_vertices_list) [source]

Given input sequences, and number or correct vertices in each of them, build an ILP representation of the problem.

Parameters:

- sequences_list – List of sequences, each of length self.m, of integers between 0 and 9.
- n_correct_vertices_list – Number of correct vertices in each sequence in sequences_list. A vertex is correct if its color is equal to the color of the corresponding vertex in the solution s_star.

Returns:

tuple, containing:

- ilp_solver: Pulp instance, holding all the information needed for the solution.
- s_star_to_color_edges: The edges (variables) in the ilp_solver.

solve_ilp(ilp_solver, s_star_to_color_edges) [source]

Given a solver with the needed information, solve the ILP and extract the solution to problem 185.

Parameters:

- ilp_solver – Pulp instance, holding all the information needed for the solution.
- s_star_to_color_edges – The edges (variables) in the ilp_solver.

Returns:

s_star: List of integers that is the solution to problem 185.

'sphinx.ext.autodoc'

```
# --  
# You  
exte  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your doctstring,  
# it will appear in a highlighted box in the documentation  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# wcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx INHO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autodoc_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{230,238,238}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/sphinx-doc/sphinx-themes-graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

autoclass_content = "both"

Include both the class's and
the init's docstrings

```
'sphinx.ext.autodoc'

# -- Extensions configuration and configurations.

# You can add here whatever extensions you want to use.
# exts = []

# Above extensions explanation and configurations.

# todo: When you use the syntax ".todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation.
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# You can click on it to see the original code.
# You can checkout the available viewcode styles here: https://help.readthedocs.org/viewcode.html
viewcode_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It generates documentation for the docstrings in your code.
# See https://github.com/sphinx-doc/sphinx/blob/master/doc/conf.py#L11-L14
# master_doc = 'index'

# payments: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{230,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...

# -- Options for HTML output

# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/rtfd/sphinx-themes/tree/graphite
# Some adjustments I made to graphite:
# - I did not use the pygments configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

In the documentation, keep
the same order of members as
in the code

`autodoc_member_order = 'bysource'`

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
  
# The init's docstrings.  
# keep the same order of members as in the code.  
# docstrings of all the class/module members.  
# but not directly. It is first rendered to an image.  
# to your LaTeX document.  
  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{(230,238,238)}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/CaiqueSantos/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf  
# and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

Default: include the docstrings of all the class/module members

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It's a plugin that automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# -- Options for LaTeX output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/sphinx-doc/sphinx/tree/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf  
# - I removed the 'graphite' theme from the static folder, and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to the project directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{math}{ofwhite}{rgb}{238,238,238}
\everymath{\mathcal{V}math}
\everydisplay{\mathcal{V}display}
...
# -- Options for HTML output --
# The theme to use
# You can find
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
#   https://github.com/Carto/sphinx-theme-graphite
#   Some adjustments I made to graphite
#   I did not use the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#   To do this, I deleted the pygments configuration both in graphite.py and in html4css1.css,
#   names.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Options for HTML output

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

names.
ve, in the templates_path comment)
tes.

Part II - Main configuration file - conf.py



Sphinx185

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)

Quick search

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
#           It will automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages
# You can find them at: https://github.com/rtfd/sphinx-themes-graphite
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/Carto/sphinx-theme-graphite
# Some adjustments I made to graphite:
# I did not use the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# To do this, I deleted the pygments configuration both in graphite.py and in html4css1.css,
```

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax "... todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.
```

```
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
# -- Options for HTML output -----
```

```
# The theme to use
# You can find more themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/Carto/sphinx-theme-graphite
# Some adjustments I made to graphite:
# I did not use the pygments configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# To do this, I deleted the pygments configuration both in graphite.py and in html4css1.css,
```

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Sphinx185

Options for HTML output

Part II - Main configuration file - conf.py

copy `<Sphinx install dir>/themes/basic/globaltoc.html` into the `'_templates` folder,
and replaced 'Table of Content' with 'Sphinx185':

Sphinx185

```
<h3>
    <a href="{{ pathto(master_doc) }}>
        {{ _('Sphinx185') }}</a>
</h3>
{{ toctree() }}
```

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configuration:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
..."""

# -- Options for HTML output -----
# The theme to use for HTML and HTML Help pages
# You can find them at: https://github.com/Carto/sphinx-theme-graphite
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/Carto/sphinx-theme-graphite
# Some adjustments I made to graphite:
# I did not use the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# To do this, I deleted the "graphite" theme, and then copied the relevant files from graphite.py and html4css1.css,
# into my own theme directory, and renamed them both to graphite.py and html4css1.css,
```

Options for HTML output

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

names.
ve, in the templates_path comment)
tes.

Part III - Document YOUR next project

Part III - Document YOUR next project

We will follow these instructions:

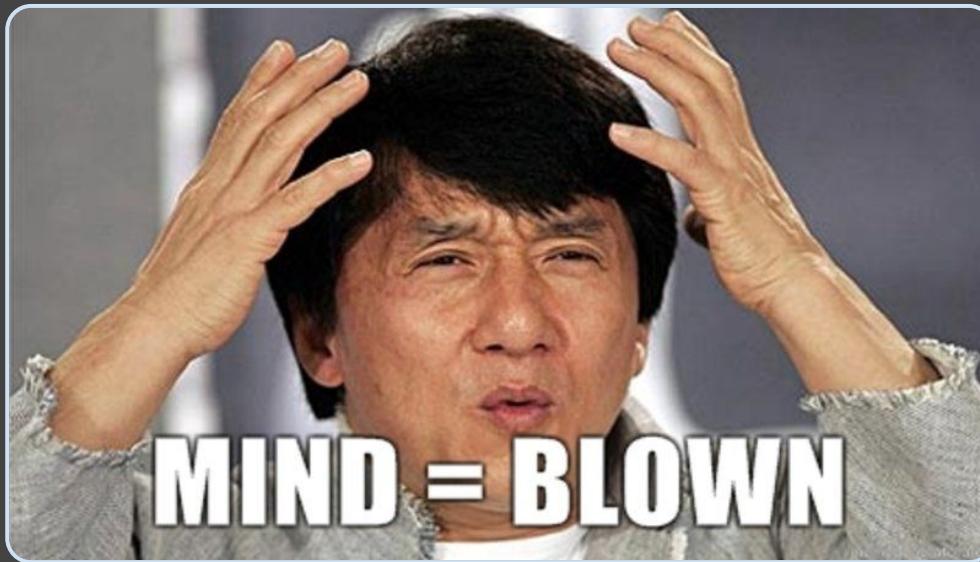
(found here https://dalyag.github.io/Sphinx185/how_to_use_this_for_your_next_project.html)

How To Use This for YOUR Next Project :)

1. `pip install sphinx`
2. Copy the `documentation_template_for_your_next_project` folder in this repository, make it a subdirectory in your local project and rename it `documentation`.
3. Edit `conf.py` by searching the pattern `#CHNAGEME#` inside the file and follow the instructions.
4. Edit `index.rst` by following the inline instructions.
5. To add a page to your documentation:
 - a. Create a `.rst` file for the function/module you wish to document (you can use the templates supplied here), and
 - b. Add the name of the `.rst` file to the `toctree` in `index.rst`.
6. In terminal, inside the `documentation` folder, run `make html`.
(TIP OF THE WEEK: actually, always run `make clean html` to clear sphinx cache and build from scratch)
7. To view locally - open the file `documentation/_build/html/index.html` in your browser, and enjoy the read :)
8. To share - you can use [GitHub Pages](#) to host your documentation:
 - a. Copy the content of `documentation/_build/html/` into a new `docs` folder, under the root of the project.
 - b. Create an empty file `_nojekyll` inside `docs` folder (this tells GitHub Pages to bypass the default `jekyll` themes and use the `html` and `css` in your project)
 - c. Push your changes to master branch.
 - d. In your repository on GitHub, go to "Settings" -> "GitHub Pages" -> "Source" and select "master branch /docs folder".
 - e. Share your beautiful documentation site at https://<your_git_usrname>.github.io/<project_name>/

Part III - Document YOUR next project

1. pip install sphinx

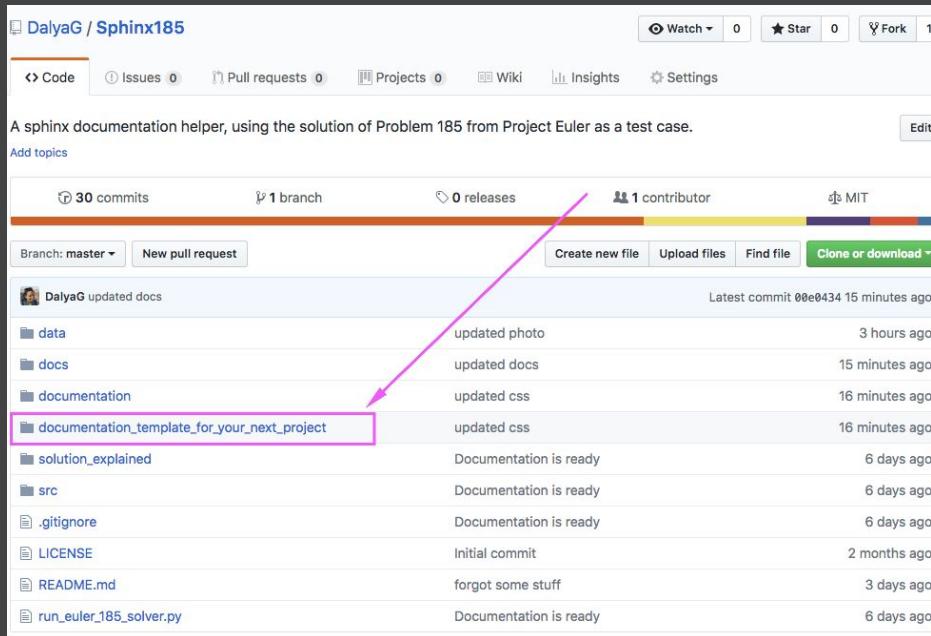


Part III - Document YOUR next project

2. Go to <https://github.com/DalyaG/Sphinx185>

Part III - Document YOUR next project

2. Go to <https://github.com/DalyaG/Sphinx185>



Part III - Document YOUR next project

2. Go to <https://github.com/DalyaG/Sphinx185>

- * Download this folder to your local project:
documentation_template_for_your_next_project
- * Rename it *documentation*

Part III - Document YOUR next project

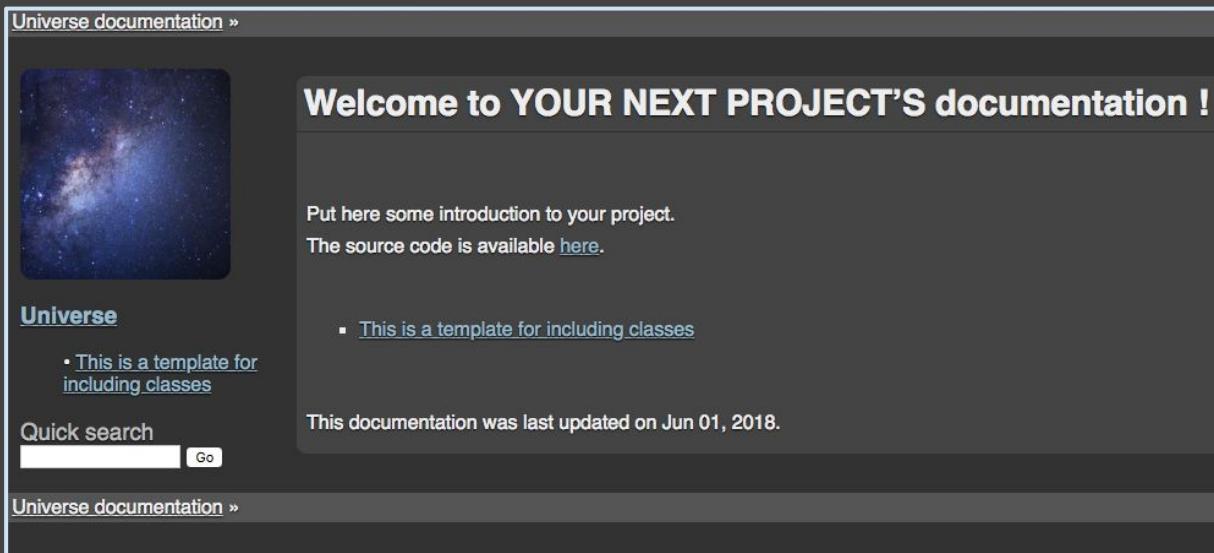
Bravo! You now have a documentation!

Part III - Document YOUR next project

Bravo! You now have a documentation!

Open this file in your browser to see:

<your_project>/documentation/_build/html/index.html



Part III - Document YOUR next project

3. Edit *conf.py*

Part III - Document YOUR next project

3. Edit *conf.py* by searching the pattern `#CHNAGEME#` inside the file and follow the instructions.

```
# --- Some general info about the project ---

#CHNAGEME# Change this to fit your project.
project = u'Universe'
copyright = u'2018, DalyaG'
author = u'DalyaG'
```

Part III - Document YOUR next project

4. Edit *index.rst*

Part III - Document YOUR next project

4. Edit *index.rst* by following the inline instructions.

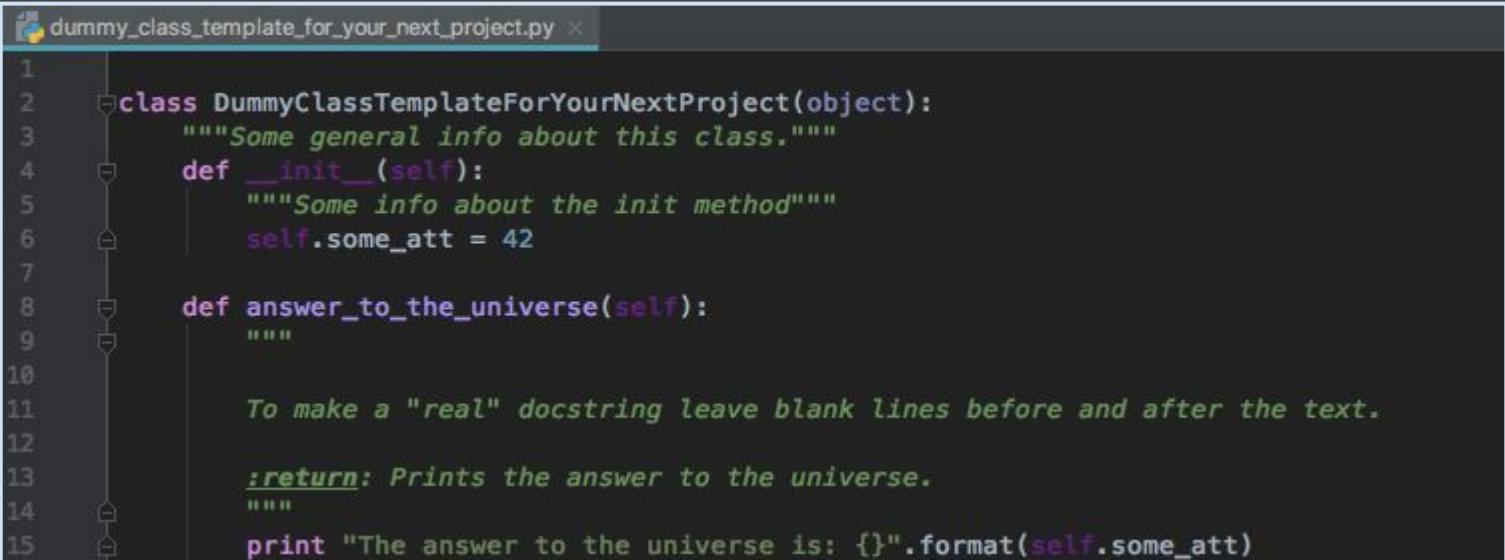
```
Welcome to YOUR NEXT PROJECT'S documentation !
=====
| Put here some introduction to your project.
| The source code is available `here <https://github.com/username/projectname>`_.
```

Part III - Document YOUR next project

5. To add a page to your documentation:

Part III - Document YOUR next project

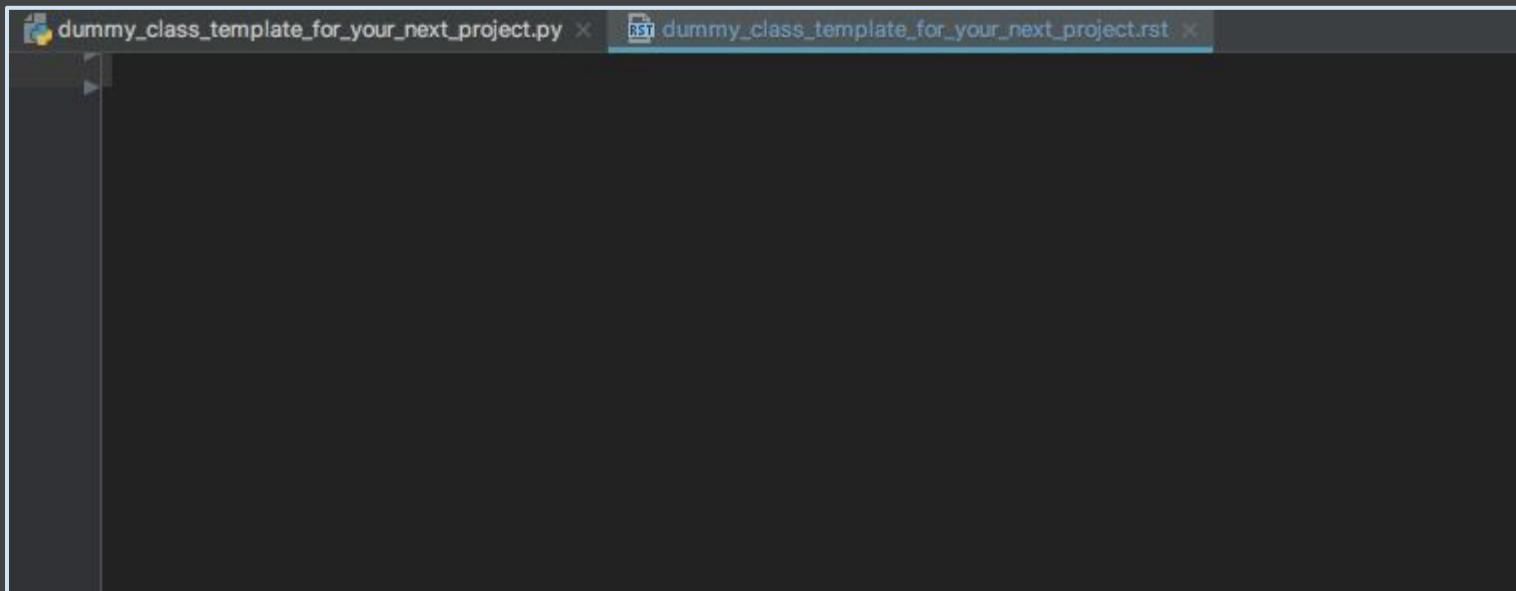
5. To add a page to your documentation:



```
1  dummy_class_template_for_your_next_project.py ×
2
3  class DummyClassTemplateForYourNextProject(object):
4      """Some general info about this class."""
5      def __init__(self):
6          """Some info about the init method"""
7          self.some_att = 42
8
8      def answer_to_the_universe(self):
9          """
10
11          To make a "real" docstring leave blank lines before and after the text.
12
13          :return: Prints the answer to the universe.
14          """
15          print "The answer to the universe is: {}".format(self.some_att)
```

Part III - Document YOUR next project

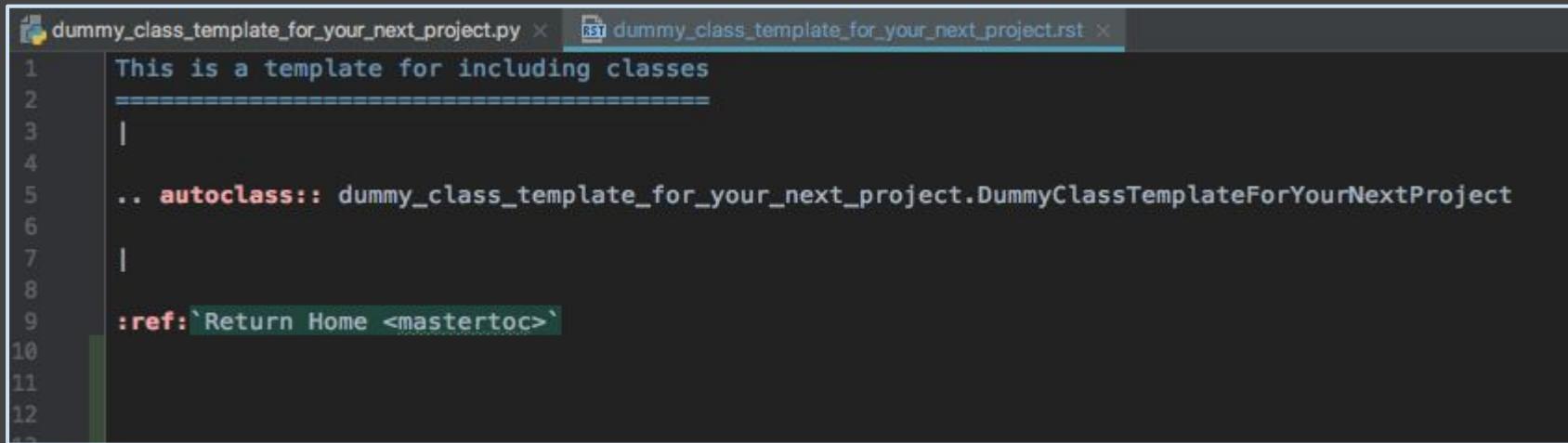
5. To add a page to your documentation:
Create a *.rst* file



Part III - Document YOUR next project

5. To add a page to your documentation:

Create a `.rst` file and edit is to include your module



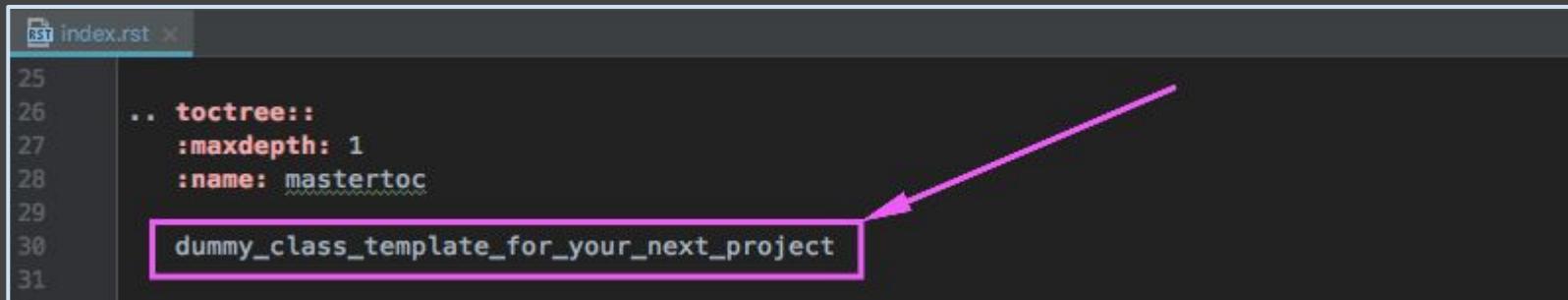
The screenshot shows a code editor with two tabs open. The left tab is named `dummy_class_template_for_your_next_project.py` and contains Python code. The right tab is named `RST dummy_class_template_for_your_next_project.rst` and contains reStructuredText (rst) code.

```
This is a template for including classes
=====
|
.. autoclass:: dummy_class_template_for_your_next_project.DummyClassTemplateForYourNextProject
|
:ref:`Return Home <mastertoc>`
```

Part III - Document YOUR next project

5. To add a page to your documentation:

Create a `.rst` file and edit it to include your module
And add its name to the *TOC* in `index.rst`



```
index.rst
25
26 .. toctree::
27     :maxdepth: 1
28     :name: mastertoc
29
30     dummy_class_template_for_your_next_project
31
```

A screenshot of a code editor showing the file `index.rst`. The code contains reStructuredText directives. A pink rectangular box highlights the line `dummy_class_template_for_your_next_project`, and a pink arrow points from the bottom right towards this highlighted line.

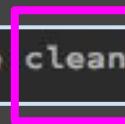
Part III - Document YOUR next project

6. In terminal, inside the *documentation* folder, run *make clean html*:

Part III - Document YOUR next project

6. In terminal, inside the *documentation* folder, run *make clean html*:

```
~/Documents/Sphinx185/documentation (git::master) ] make clean html
```

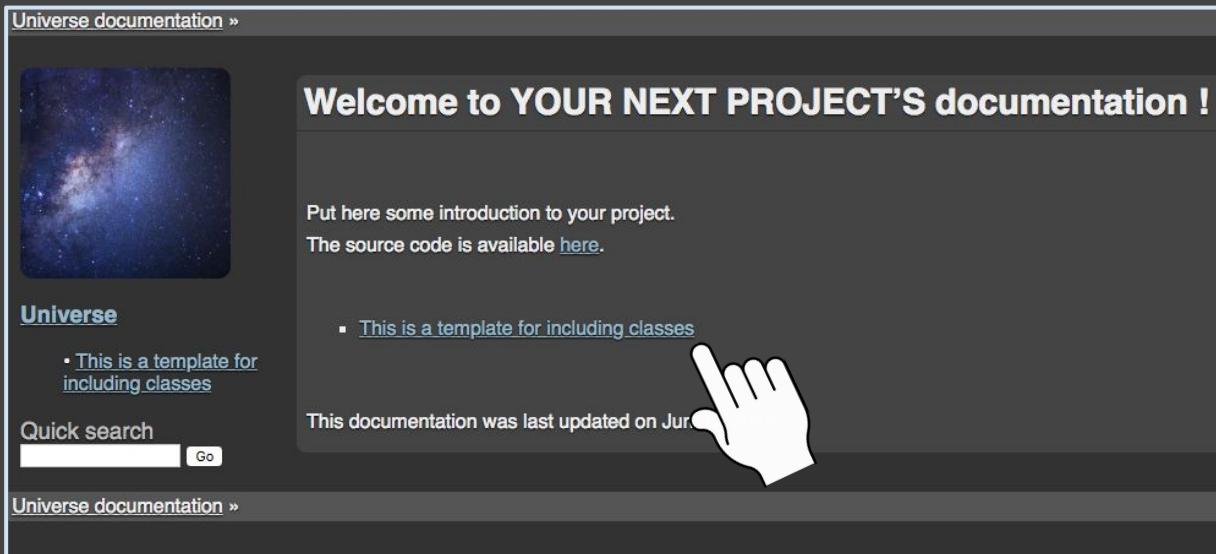


Part III - Document YOUR next project

7. To view locally

Part III - Document YOUR next project

7. To view locally, open this file in your browser
documentation/_build/html/index.html



The screenshot shows a web browser displaying a documentation page. The title bar reads "Universe documentation »". The main content area features a large image of a galaxy on the left and a bold title "Welcome to YOUR NEXT PROJECT'S documentation !" on the right. Below the title, there is placeholder text: "Put here some introduction to your project." and "The source code is available [here](#)". A sidebar on the left is titled "Universe" and contains a bullet point: "• This is a template for including classes". At the bottom of the page, there is a "Quick search" input field and a "Go" button. A large white hand icon with a pointing finger is overlaid on the bottom right corner of the page content.

Universe documentation »

Welcome to YOUR NEXT PROJECT'S documentation !

Put here some introduction to your project.
The source code is available [here](#).

Universe

- This is a template for including classes

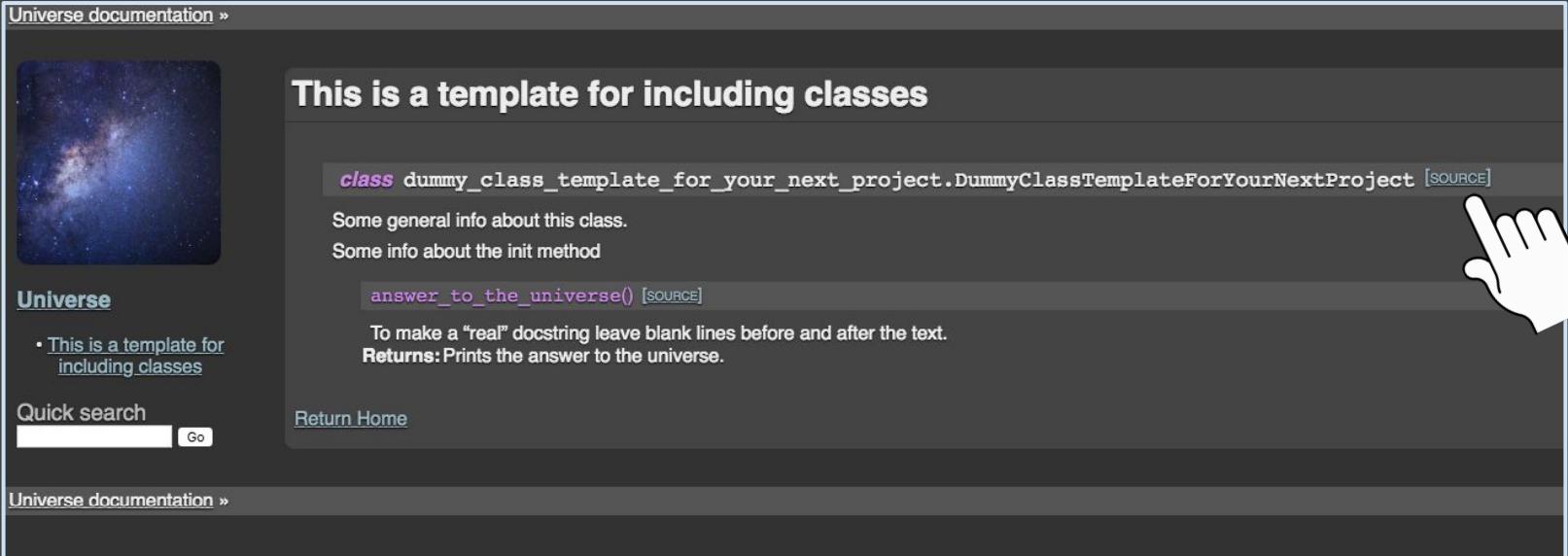
Quick search Go

This documentation was last updated on Jun 1, 2023.

Universe documentation »

Part III - Document YOUR next project

7. To view locally, open this file in your browser
documentation/_build/html/index.html



Universe documentation »

This is a template for including classes

`class dummy_class_template_for_your_next_project.DummyClassTemplateForYourNextProject [SOURCE]`

Some general info about this class.
Some info about the init method

`answer_to_the_universe() [SOURCE]`

To make a “real” docstring leave blank lines before and after the text.
Returns: Prints the answer to the universe.

Universe

- This is a template for including classes

Quick search Go

Return Home

Universe documentation »

Part III - Document YOUR next project

7. To view locally, open this file in your browser *documentation/_build/html/index.html*

Universe documentation » Module code »



Source code for dummy_class_template_for_your_next_project

```
[docs]class DummyClassTemplateForYourNextProject(object):
    """Some general info about this class."""
    def __init__(self):
        """Some info about the init method"""
        self.some_att = 42

[docs]    def answer_to_the_universe(self):
        """
        To make a "real" docstring leave blank lines before and after the text.

        :return: Prints the answer to the universe.
        """
        print "The answer to the universe is: {}".format(self.some_att)
```

Universe

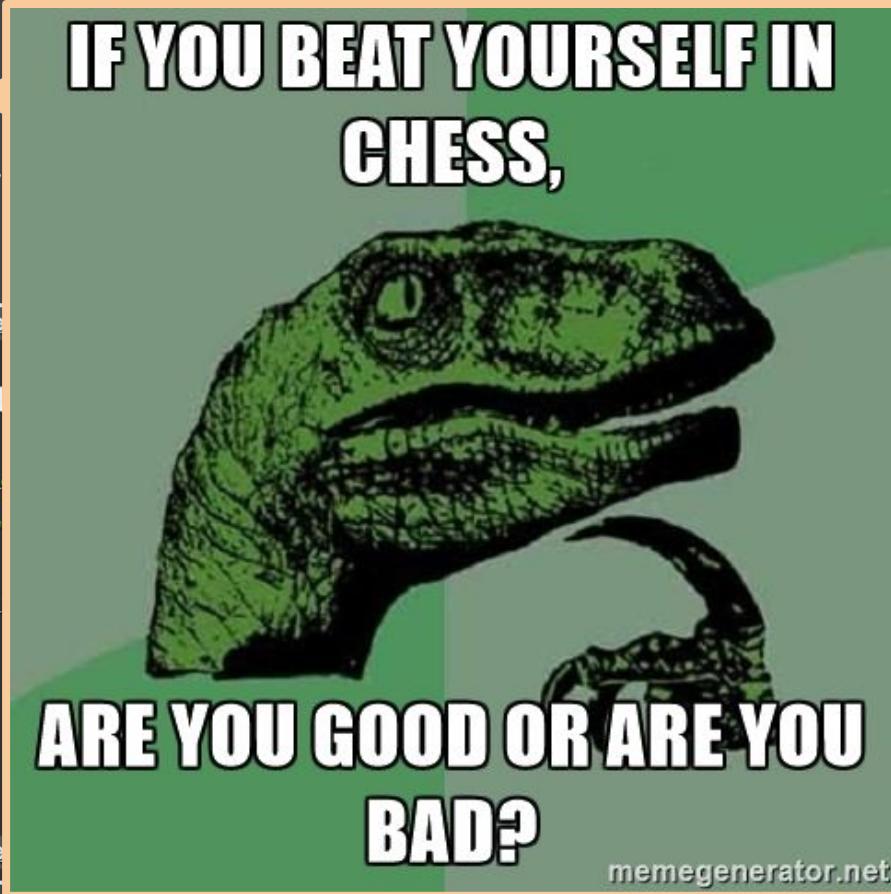
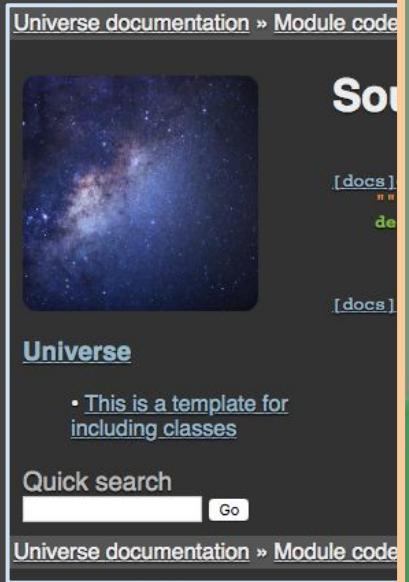
- This is a template for including classes

Quick search Go

Universe documentation » Module code »

Part III - Document YOUR next project

7. To view local documentation



browser

your_next_project

Part III - Document YOUR next project

8. SHARE your documentation!



SHARING IS CARING

Part III - Document YOUR next project

8. Hosting on GitHub Pages:



Part III - Document YOUR next project

8. Hosting on GitHub Pages:

Part III - Document YOUR next project

8. Hosting on GitHub Pages:

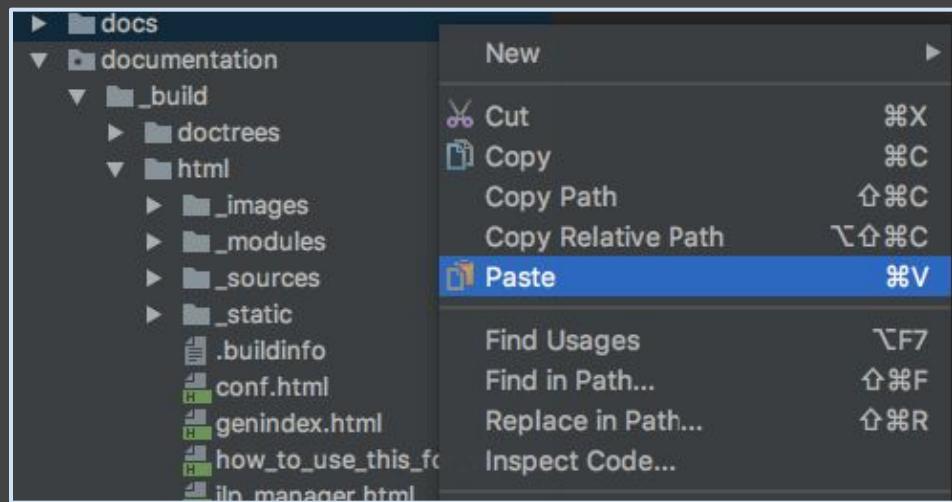
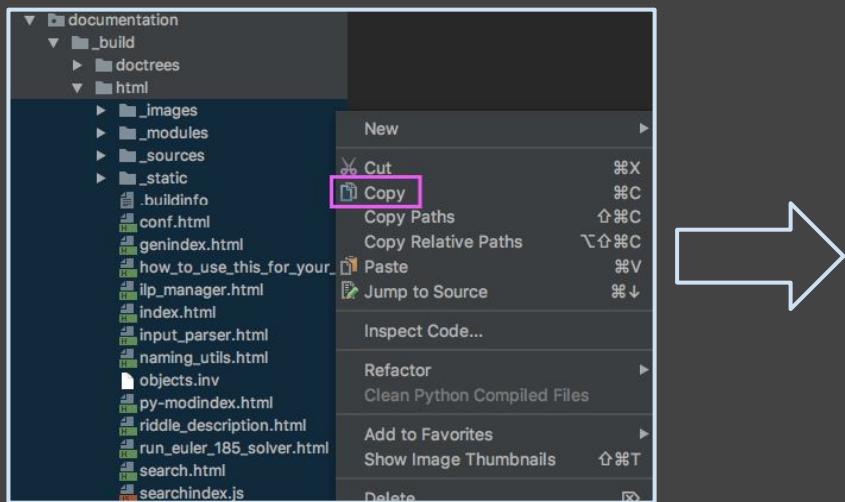
a. Create a folder *docs* under the root of the project



Part III - Document YOUR next project

8. Hosting on GitHub Pages:

a. And copy inside it the content of
documentation/_build/html/

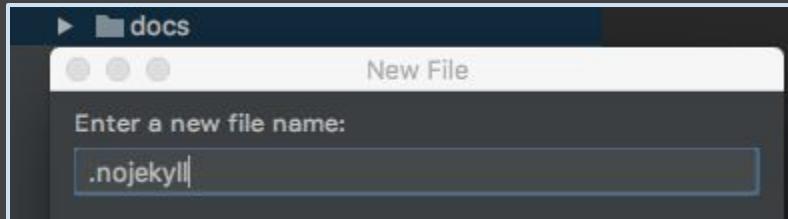


Part III - Document YOUR next project

8. Hosting on GitHub Pages:

b. Create an empty file `.nojekyll` inside `docs` folder

(this tells GitHub Pages to bypass the default jekyll themes and use the html and css in your project)



Part III - Document YOUR next project

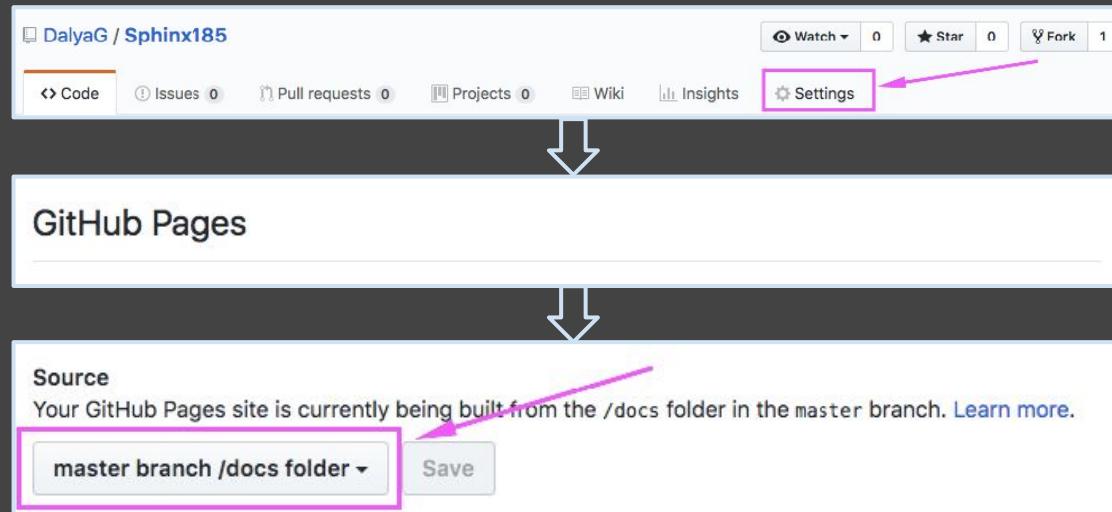
8. Hosting on GitHub Pages:

c. Push your changes to master branch.

Part III - Document YOUR next project

8. Hosting on GitHub Pages:

d. In your repository on GitHub, go to “Settings” -> “GitHub Pages” -> “Source”



Part III - Document YOUR next project

8. Hosting on GitHub Pages:

e. Share your beautiful documentation site at

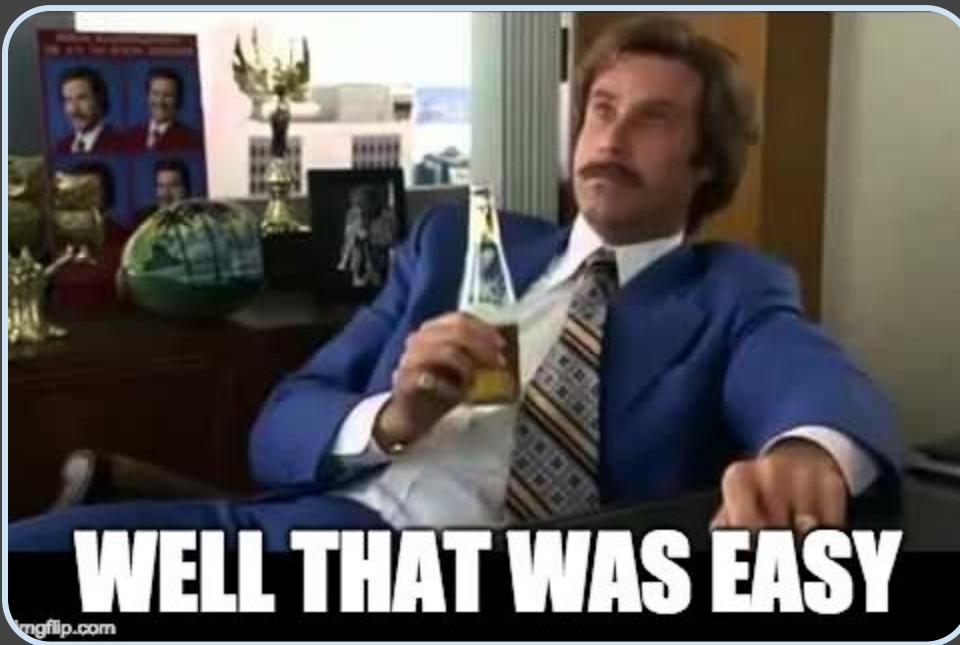
https://<your_git_username>.github.io/<project_name>/

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

✓ Your site is published at <https://dalyag.github.io/Sphinx185/>

Part III - Document YOUR next project



Part III - Document YOUR next project

WHO CAN DO IT?



YOU CAN DO IT!

Take Home Message



Thank you :)

Questions?

DalyaG@gmail.com



EXTRA SLIDES

Intializing sphinx

Part III - Document YOUR next project

2. Initialize sphinx: there is always the option of

```
~/Documents/Sphinx185 (git::master) ] sphinx-quickstart
```

Part III - Document YOUR next project

2. Initialize sphinx: there is always the option of

```
~/Documents/Sphinx185 (git::master) ] sphinx-quickstart
```

One advice: create a dedicated documentation folder

```
Enter the root path for documentation.
```

```
> Root path for the documentation [.]: documentation
```

Part III - Document YOUR next project

2. Initialize sphinx: there is always the option of

```
~/Documents/Sphinx185 (git::master) ] sphinx-quickstart
```

One advice: create a dedicated documentation folder

```
Enter the root path for documentation.
```

```
> Root path for the documentation [.]: documentation
```

After that its all Blah blah blah, enter enter enter...

Main configuration file - conf.py

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-

# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory --

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project --

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

```
# -- Define and configure non-default extensions ----

# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax ``todo`` some todo command`` in your doctstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
# It will automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to add to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{color}
\definecolor{offwhite}{rgb}{230,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
'''

# -- Options for HTML output --

# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the `graphite` template from here:
# https://github.com/sphinx-doc/sphinx-themes-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_dalya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In this project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory ----

import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project ----

project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations ----

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
# templates_path = ['_templates']
```

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.
```

```
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))
```

```
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar.
# So I copied `<Sphinx install dir>/themes/basic/globaltoc.html` into the `_templates` folder,
# and replaced `Table of Content` with `Sphinx185`.
# templates_path = ['_templates']
```

Let Sphinx know what is
the path to your project

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-

# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# project = u'Sphinx185'
# copyright = u'2018, DalyaG'
# author = u'DalyaG'

# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table of Contents in the sidebar.
# So I copied `<Sphinx install dir>/themes/basic/globaltoc.html` into the `_templates` folder,
# and replaced `Table of Content` with `Sphinx185`.
# templates_path = ['_templates']
```

Some basic info about
the project. This will
appear in the
documentation.

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory -----
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project -----
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'
```

source_suffix = ['.rst']

```
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/templating.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar,
# So I copied `<Sphinx install dir>/themes/basic/globaltoc.html` into the `_templates` folder,
# and replaced `Table of Content` with `Sphinx185`.
# templates_path = ['_templates']
```

What markdown languages will you be using to generate pages

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory -----
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project -----
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'
```

source_suffix = ['.rst']

```
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right :)
# So, for example, in this project, I wanted to change the title of the Table Of Contents in the sidebar,
# So I copied `<Sphinx install dir>/themes/basic/globaltoc.html` into the `_templates` folder,
# and replaced `Table of Content` with `Sphinx185`.
templates_path = ['_templates']
```

What markdown languages will you be using to generate pages

.rst = reStructuredText

<http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html>

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory --
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project --
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']

# List here any paths that contain templates, relative to this directory.
# You can find some not-so-intuitive information here: http://www.sphinx-doc.org/en/master/template.html
# But the best way to learn is by example, right? :)
# So, for example, in this project, I wanted to change the title of the Table of Contents in the sidebar.
# So I copied '<Sphinx install dir>/themes/basic/globaltoc.html' into the '_templates' folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

master_doc = 'index'

What is the name of the master document (landing page)

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-

# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory --
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project --
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an 'index.rst' file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: '_build/html/index.html'
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# So I copied <Sphinx install dir>/themes/basic/_globaltoc.html into the _templates folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

Inside '`_build`' is the output of the documentation.

`exclude_patterns = ['_build']`

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-

# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory --
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project --
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations --
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an 'index.rst' file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: '_build/html/index.html'
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# So I copied <Sphinx install dir>/themes/basic/_globaltoc.html into the _templates folder,
# and replaced 'Table of Content' with 'Sphinx185'.
templates_path = ['_templates']
```

Inside '`_build`' is the output of the documentation.

And we don't want to recursively document what we recursively document what we...

`exclude_patterns = ['_build']`

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory -----
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project -----
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations -----
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']
```

Templates are html
examples for designing
predifined elements

templates_path = ['_templates']

Part II - Main configuration file - conf.py

```
"""
This is the main file in which the configuration for the documentation is made.

"""

# -*- coding: utf-8 -*-
#
# Sphinx185 documentation build configuration file, created by
# sphinx-quickstart on Sat May 12 19:34:31 2018.

# DalyaG: This file was heavily modified from its original build.
# Hope you find this useful :)

# -- Define here your working directory -----
import os
import sys
sys.path.append(os.path.abspath('/Users/dalya/Documents/Sphinx185'))

# -- Some general info about the project -----
project = u'Sphinx185'
copyright = u'2018, DalyaG'
author = u'dalyaG'

# -- A few basic configurations -----
# The documentation in this project will be mostly generated from .rst files
# In this project, every auto-documented module/class has its own .rst file, under the main documentation dir,
# which is rendered into an .html page.
# Get more information here: http://www.sphinx-doc.org/en/master/usage/restructuredtext/basics.html
source_suffix = ['.rst']

# This is the name of the main page of the project.
# It means that you need to have an `index.rst` file where you will design the landing page of your project.
# It will be rendered into an .html page that you can find at: `build/html/index.html`
# (this is a standard name. change it only if you know what you are doing)
master_doc = 'index'

# List of patterns, relative to source directory, that match files and
# directories to ignore when looking for source files.
# This patterns also effect to html_static_path and html_extra_path
exclude_patterns = ['_build']
```

Templates are html
examples for designing
predifined elements

We will see:
a template for sidebar

templates_path = ['_templates']

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax "... todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/sphinx-doc/sphinx-themes-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# and deleted graphite.py
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
# you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc',  
'sphinx.ext.imgmath']
```

```
#           It will appear in a highlighted box in the documentation.  
#           In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
#           The source code will have colors defined by the Pygments (syntax highlighting) style.  
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
#           It allows you to automatically generate documentation for the docstrings in your code.  
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{230,238,238}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/sphinx-doc/sphinx/tree/graphite  
# Some adjustments I made to graphite:  
#   - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
#     you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo.)  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
#   - Table of Contents: I chose globaltoc as it is less refined,  
#     and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
#   - Searchbox: appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Let Sphinx know what extensions to use

Part II - Main configuration file - conf.py

'sphinx.ext.todo'

```
# todo: When you use the syntax "... todo: some todo comment" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/CookP/sphinx-theme-graphite
# Some adjustments I made to graphite:
#   - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#   - I copied the graphite.css and deleted graphite.py
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#     you can compare the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#   and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#   - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

'sphinx.ext.todo'

```
# todo: When you use the syntax "... todo:: some todo comment" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It will automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''
```

.. todo:: find a better name for n_correct_vertices_list



```
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/Cehoy/Sphinx-theme-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']
```

```
static files (such as style sheets) here,
are copied after the builtin static files,
overwrote the builtin 'default.css'.
```

```
to add my own logo for the project:
ice support the use of logo.
.png
```

```
dictionary that maps document names to template names.
in the sidebar;
also as it is less refined,
the globaltoc template (see explanation above, in the templates_path comment)
OC, and can be configured by editing css attributes.
```

Todo

find a better name for n_correct_vertices_list

Part II - Main configuration file - conf.py

'sphinx.ext.todo'

todo_include_todos = True

```
# coding: utf-8
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''
```

.. todo:: find a better name for n_correct_vertices_list



```
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/CaiqueSantos/graphite
# Some adjustments I made to graphite:
# - I did not use the pygments configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']
```

Todo

find a better name for n_correct_vertices_list

Part II - Main configuration file - conf.py

'sphinx.ext.viewcode'

```
# Above extensions explanation and configurations.

# todo: When you use the syntax ".todo", some "todo command" in your docstring,
#       it will appear in a highlighted box in the documentation.
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{(230,238,238)}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite` template from here:
# https://github.com/CaiqueSantos/graphite
# Some adjustments I made to graphite:
#   - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#   - I removed the graphite.css and deleted graphite.py
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#     you can compare the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#   and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#   - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

'sphinx.ext.viewcode'

```
# Above extensions explanation and configurations.

# todo: When you use the syntax ".todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''

    \begin{array}{c} \text{imgmath} \\ \text{imgmath} \end{array}
```

src.input_parser.input_parser(m) [SOURCE]



```
[docs]def input_parser(m):
    """
    Given length of sequences, load input data corresponding to this length.

    :param m: Length of sequences in the input.
    :return: tuple, containing:
        1. sequences_list: List of sequences in the input, parsed such that \
                           sequences_list[i] holds a list of integers that are the colors of this sequence.
        2. n_correct_vertices_list: List holding the number of correct vertices in each sequence in sequences_list.

    .. note:: This function assumes the existence of 'data/input_m.txt'

    .. todo:: find a better name for n_correct_vertices_list
    """

    print "Loading input..."
    with open('data/input_{}.txt'.format(m), 'r') as f:
        data = [line.rstrip() for line in f]
    sequences_list = [[int(i) for i in item[:m]] for item in data]
    n_correct_vertices_list = [int(item[m + 2]) for item in data]
    return sequences_list, n_correct_vertices_list
```

```
# The theme I chose for HTML and HTML Help pages.
# You can change the theme to another one. See themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project I chose to use a non-default theme, and so I downloaded the 'graphite' template from here:
#   https://github.com/rtomayko/sphinx-theme-graphite
# and added it to graphite
# I also removed the payment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# and deleted graphite.py
# If you want to see what changes I made, download the original and compare to find those changes.
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#   and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#   - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

'sphinx.ext.viewcode'

pygments_style = 'native'

<https://help.farbox.com/pygments.html>

src.input_parser.input_parser(m) [SOURCE]



```
[docs]def input_parser(m):
    """
    Given length of sequences, load input data corresponding to this length.

    :param m: Length of sequences in the input.
    :return: tuple, containing:
        1. sequences_list: List of sequences in the input, parsed such that \
            sequences_list[i] holds a list of integers that are the colors of this sequence.
        2. n_correct_vertices_list: List holding the number of correct vertices in each sequence in sequences_list.

    .. note:: This function assumes the existence of 'data/input_m.txt'

    .. todo:: find a better name for n_correct_vertices_list
    """

    print "Loading input..."
    with open('data/input_{}.txt'.format(m), 'r') as f:
        data = [line.rstrip() for line in f]
    sequences_list = [[int(i) for i in item[m]] for item in data]
    n_correct_vertices_list = [int(item[m + 2]) for item in data]
    return sequences_list, n_correct_vertices_list
```

```
# The 'graphite' theme is a clean, modern theme for HTML and RTD Help pages.
# You can use it as is, or you can extend it by creating your own themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/rtfd/graphite-theme-graphite
# and modified it to graphite
# I removed the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# and deleted graphite.py
# In the meantime, I configured several classes both in graphite.css and in html4css1.css,
# so if you want to see what I did, download the original and compare to find those changes.
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to the project directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In this project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

'sphinx.ext.autodoc'

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{(230,238,238)}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/rtfd/sphinx-themes/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
# - I copied the graphite.css and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can compare the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

.. autoclass:: src.ilp_manager.ILPManager



class src.ilp_manager.ILPManager(m) [source]

Model the 185th problem in Project Euler as an ILP (Integer Linear Program)

To instantiate this module, please specify the length on sequences.

build_ilp_solver(sequences_list, n_correct_vertices_list) [source]

Given input sequences, and number or correct vertices in each of them, build an ILP representation of the problem.

Parameters:

- sequences_list – List of sequences, each of length self.m, of integers between 0 and 9.
- n_correct_vertices_list – Number of correct vertices in each sequence in sequences_list. A vertex is correct if its color is equal to the color of the corresponding vertex in the solution s_star.

Returns:

tuple, containing:

- ilp_solver: Pulp instance, holding all the information needed for the solution.
- s_star_to_color_edges: The edges (variables) in the ilp_solver.

solve_ilp(ilp_solver, s_star_to_color_edges) [source]

Given a solver with the needed information, solve the ILP and extract the solution to problem 185.

Parameters:

- ilp_solver – Pulp instance, holding all the information needed for the solution.
- s_star_to_color_edges – The edges (variables) in the ilp_solver.

Returns:

s_star: List of integers that is the solution to problem 185.

'sphinx.ext.autodoc'

```
# --#
# You
exte
# Above extensions explanation and configurations.

# todo: When you use the syntax ".todo: some todo command" in your doctstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autodoc_class_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\def\mathcolor[rgb]{230,230,230}{\color{offwhite}}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/sphinx-doc/sphinx-themes-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/epiphany_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Content box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

autoclass_content = "both"

Include both the class's and
the init's docstrings

```
'sphinx.ext.autodoc'

# -- Extensions configuration and configurations.

# You can add here whatever extensions you want to use.
# exts = []

# Above extensions explanation and configurations.

# todo: When you use the syntax ".todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation.
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# You can click on it to see the original code.
# You can checkout the available viewcode styles here: https://help.readthedocs.org/viewcode.html
viewcode_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It generates documentation for the docstrings in your code.
# See https://github.com/sphinx-doc/sphinx/blob/master/doc/conf.py#L11-L14
# master_doc = 'index'

# payments: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{color}
\definecolor{offwhite}{rgb}{230,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...

# -- Options for HTML output

# The theme to use for HTML and HTML Help pages.
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:
# https://github.com/rtfd/sphinx-themes/tree/graphite
# Some adjustments I made to graphite:
# - I did not use the pygments configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#   you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Search box: appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

In the documentation, keep
the same order of members as
in the code

`autodoc_member_order = 'bysource'`

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
  
# The init's docstrings.  
# keep the same order of members as in the code.  
# docstrings of all the class/module members.  
# but not directly. It is first rendered to an image.  
# to your LaTeX document.  
  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{(230,238,238)}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/CaiqueSantos/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf  
# and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

Default: include the docstrings of all the class/module members

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo: some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It's a plugin that automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# -- Options for LaTeX output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/sphinx-doc/sphinx/tree/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf  
# - I removed the 'graphite' theme from the static folder, and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to the project directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo; some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{color}  
\\definecolor{offwhite}{rgb}{(230,238,238)}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/rtomayko/sphinx-theme-graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
# - I copied the graphite.css file and deleted graphite.py  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
# you can compare the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['.']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['_static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo).  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
# and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# Content box appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

Let `:math:`\color{red}s^{\star}`` be a sequence



Let `s*` be a sequence

```
# --  
# You  
exte  
  
# Above extensions explanation and configurations:  
  
# todo: When you use the syntax ".todo; some todo command" in your docstring,  
# it will appear in a highlighted box in the documentation.  
# In order for this extension to work, make sure you include the following:  
todo_include_todos = True  
  
# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.  
# The source code will have colors defined by the Pygments (syntax highlighting) style.  
# You can checkout the available pygments here: https://help.farbox.com/pygments.html  
pygments_style = 'native'  
  
# autodoc: The best thing about Sphinx IMO is autodoc.  
# It allows you to automatically generate documentation for the docstrings in your code.  
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html  
# Some useful configurations:  
autoclass_content = "both" # Include both the class's and the init's docstrings.  
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.  
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.  
  
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.  
# You can add here whatever preamble you are used to adding to your LaTeX document.  
imgmath_latex_preamble = r'''  
\\usepackage{xcolor}  
\\definecolor{offwhite}{rgb}{230,238,238}  
\\everymath{\\color{offwhite}}  
\\everydisplay{\\color{offwhite}}  
...  
  
# -- Options for HTML output --  
  
# The theme to use for HTML and HTML Help pages.  
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html  
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here:  
# https://github.com/sphinx-doc/sphinx/tree/graphite  
# Some adjustments I made to graphite:  
# - I did not use the pygment configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf  
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,  
#   you can download the original and compare to find those changes.  
html_theme = 'graphite'  
# When using a non-built-in theme, define the path to your template code.  
html_theme_path = ['']  
  
# Add any paths that contain custom static files (such as style sheets) here,  
# relative to this directory. They are copied after the builtin static files,  
# so a file named 'default.css' will overwrite the builtin 'default.css'.  
html_static_path = ['static']  
  
# Defining the static path allows me to add my own logo for the project:  
# (make sure the theme of your choice support the use of logo.)  
html_logo = '_static/sphinx_and_daiya.png'  
  
# Custom sidebar templates, must be a dictionary that maps document names to template names.  
# In This project I chose to include in the sidebar:  
# - Table of Contents: I chose globaltoc as it is less refined,  
#   and I configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)  
# - Search box: appears below the ToC, and can be configured by editing css attributes.  
html_sidebars = {  
    '**': [  
        'globaltoc.html',  
        'searchbox.html'  
    ]  
}
```

Part II - Main configuration file - conf.py

Let :math:`\color{red}s^*` be a sequence



Let s^* be a sequence

```
# ...
# You
exte
# Above extensions explanation and configurations:
# todo: When you use the syntax ".todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.karbox.com/pygments.html
pygments_style = 'native'

imgmath_latex_preamble = r'''
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
'''
```

ings in your code.
members as in the code.
lass/module members.
It is first rendered to an image.

```
# You can find available themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the 'graphite' template from here;
#   https://github.com/Carcinogen/sphinx-theme-graphite
#   Some adjustments I made to graphite:
#     - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#     - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#       you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
#   (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#     and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#     Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

'sphinx.ext.imgmath'

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{math}{ofwhite}{rgb}{238,238,238}
\everymath{\mathcal{V}\mathbf{e}\mathit{r}\mathbf{y}\mathit{d}\mathit{is}\mathit{p}\mathit{l}\mathit{a}\mathit{y}}
\mathbf{...}

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.  See the
# themes.html theme for more information on how to
# choose a theme.
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/CarterC/sphinx-theme-graphite
# Some adjustments I made to graphite:
# - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
# you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['.']

# Add any paths that contain custom static files (such as style sheets) here,
# relative to this directory. They are copied after the builtin static files,
# so a file named 'default.css' will overwrite the builtin 'default.css'.
html_static_path = ['_static']

# Defining the static path allows me to add my own logo for the project:
# (make sure the theme of your choice support the use of logo.
html_logo = '_static/sphinx_and_daiya.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# Content box appears below the ToC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Options for HTML output

Part II - Main configuration file - conf.py

The themes holds the design
for the output html
documentation.

I downloaded ‘graphite’ and
made some personalizations.

<https://github.com/Cartroo/sphinx-theme-graphite>

```
html_theme = 'graphite'
```

```
# When using a non-built-in theme
html_theme_path = ['..']
```

```
# Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configuration:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = 'both' # Include both the class's and the init's docstrings.
autodoc_member_order = 'bysource' # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r'''
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\textcolor{offwhite}}
\everydisplay{\textcolor{offwhite}}
...
'''

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages
# You can find them at:
# https://github.com/Cartroo/sphinx-theme-graphite
# Some adjustments I made to graphite
# I did not use the human configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# and deleted graphite.py
# I added several classes both in graphite.css and in html4css1.css, and compare to find those changes.

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
# - Table of Contents: I chose globaltoc as it is less refined,
# and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
# - Content box: appears below the TOC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Options for HTML output

Part II - Main configuration file - conf.py

The folder where you keep images etc.

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']
# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # Default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\textcolor{offwhite}{#}}
\everydisplay{\textcolor{offwhite}{#}}
...
# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages.  See the
# sidebar_theme
# You can find more information about themes here:
# https://github.com/Cartesi/sphinx-theme-graphite
# Some adjustments I made to graphite:
#   - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#     you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']

# Add any paths that contain custom static files (such as style sheets) here,
# in static files,
# default.css.

# project:
html_logo = '_static/sphinx_and_django.png'

# Custom sidebar templates, must be a dictionary that maps document names to template names.
# In This project I chose to include in the sidebar:
#   - Table of Contents: I chose globaltoc as it is less refined,
#     and configured the title by editing the globaltoc template (see explanation above, in the templates_path comment)
#     Content box appears below the TOC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

Such as the logo for this project

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\textcolor{offwhite}{\color{black}\textnormal{#1}}}
\everydisplay{\textcolor{offwhite}{\color{black}\textnormal{#1}}}
"""

# -- Options for HTML output --
# The theme to use:
# You can find more information here: https://github.com/rtfd/sphinx-themes-graphite
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/rtfd/sphinx-themes-graphite
# Some adjustments I made to graphite:
#   - I did not use the pygment configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#   - In the static folder, I configured several classes both in graphite.css and in html4css1.css,
#     you can download the original and compare to find those changes.
html_theme = 'graphite'
# When using a non-built-in theme, define the path to your template code.
html_theme_path = ['..']

# Add any paths that contain custom static files (such as style sheets) here,
# in static files, relative to this directory. They are copied after the builtin static files,
# so a file named "custom.css" will overwrite the builtin "style.css".
html_static_path = ['_static']

# -- Options for HTMLHelp output -- (comment)
# Search box: appears below the TOC, and can be configured by editing css attributes.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{math}{ofwhite}{rgb}{238,238,238}
\everymath{\mathcal{V}math}
\everydisplay{\mathcal{V}display}
...
# -- Options for HTML output --
# The theme to use
# You can find
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
#   https://github.com/Carto/sphinx-theme-graphite
#   Some adjustments I made to graphite
#   I did not use the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
#   To do this, I deleted the pygments configuration both in graphite.py and in html4css1.css,
#   names.
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Options for HTML output

names.
ve, in the templates_path comment)
tes.

Part II - Main configuration file - conf.py



Sphinx185

- [Euler's 185th Riddle](#)
- [Solver Runner](#)
- [Input Parser](#)
- [Naming Utils](#)
- [The ILP Manager class](#)
- [The main configurations file: conf.py](#)

Quick search

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
"""

# -- Options for HTML output --
# The theme to use for HTML and HTML Help pages
# You can find them at: https://github.com/rtfd/sphinx-themes-graphite
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/Carto/sphinx-theme-graphite
# Some adjustments I made to graphite:
# I did not use the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# To do this, I deleted the pygments configuration both in graphite.py and in html4css1.css,
```

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Part II - Main configuration file - conf.py

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configurations:

# todo: When you use the syntax "... todo: some todo command" in your docstring,
# it will appear in a highlighted box in the documentation
# In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
# The source code will have colors defined by the Pygments (syntax highlighting) style.
# You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx IMO is autodoc.
# It allows you to automatically generate documentation for the docstrings in your code.
# Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.
```

```
# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
...
"""

# -- Options for HTML output -----
```

```
# The theme to use
# You can find more themes here: http://www.sphinx-doc.org/en/master/theming.html
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/Carto/sphinx-theme-graphite
# Some adjustments I made to graphite:
# I did not use the pygments configuration, and so removed "pygments_style = graphite.GraphiteStyle" from theme.conf
# To do this, I deleted the "pygments" section, and removed both its graphite.py and its html4css1.css,
```

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

Sphinx185

Options for HTML output

Part II - Main configuration file - conf.py

copy `<Sphinx install dir>/themes/basic/globaltoc.html` into the `'_templates` folder,
and replaced 'Table of Content' with 'Sphinx185':

Sphinx185

```
<h3>
    <a href="{{ pathto(master_doc) }}>
        {{ _('Sphinx185') }}</a>
</h3>
{{ toctree() }}
```

```
# -- Define and configure non-default extensions -----
# You can find a list of available extension here: http://www.sphinx-doc.org/en/master/extensions.html
extensions = ['sphinx.ext.todo', 'sphinx.ext.viewcode', 'sphinx.ext.autodoc', 'sphinx.ext.imgmath']

# Above extensions explanation and configuration:
# todo: When you use the syntax "... todo: some todo command" in your docstring,
#       it will appear in a highlighted box in the documentation
#       In order for this extension to work, make sure you include the following:
todo_include_todos = True

# viewcode: Next to each function/module in the documentation, you will have an internal link to the source code.
#           The source code will have colors defined by the Pygments (syntax highlighting) style.
#           You can checkout the available pygments here: https://help.farbox.com/pygments.html
pygments_style = 'native'

# autodoc: The best thing about Sphinx INHO is autodoc.
#           It allows you to automatically generate documentation for the docstrings in your code.
#           Get more info here: http://www.sphinx-doc.org/en/master/ext/autodoc.html
# Some useful configurations:
autoclass_content = "both" # Include both the class's and the init's docstrings.
autodoc_member_order = "bysource" # In the documentation, keep the same order of members as in the code.
autodoc_default_flags = ['members'] # default: include the docstrings of all the class/module members.

# imgmath: Sphinx allows use of LaTeX in the html documentation, but not directly. It is first rendered to an image.
# You can add here whatever preamble you are used to adding to your LaTeX document.
imgmath_latex_preamble = r"""
\usepackage{xcolor}
\definecolor{offwhite}{rgb}{238,238,238}
\everymath{\color{offwhite}}
\everydisplay{\color{offwhite}}
..."""

# -- Options for HTML output -----
# The theme to use for HTML and HTML Help pages
# You can find them at: https://github.com/Carto/sphinx-theme-graphite
# In this project, I wanted to use a non-default theme, and so I downloaded the "graphite" template from here:
# https://github.com/Carto/sphinx-theme-graphite
# Some adjustments I made to graphite:
# I did not use the pygments configuration, and so removed "pygments style = graphite.GraphiteStyle" from theme.conf
# To do this, I deleted the "graphite" theme, and then copied the relevant files from graphite.py and html4css1.css,
# into my own theme directory, and renamed them both to graphite.py and html4css1.css,
```

Options for HTML output

```
html_sidebars = {
    '**': [
        'globaltoc.html',
        'searchbox.html'
    ]
}
```

names.
ve, in the templates_path comment)
tes.

Some more fun features

Special boxes: note

(in the
dostring)

```
.. note:: This function assumes the existence of 'data/input_m.txt'
```



Note

This function assumes the existence of 'data/input_m.txt'

(in the
dostring)

Special boxes: exmaple

```
**Example:** ::  
  
    s_star_index_node(3) = 's_star_3'
```



Example:

```
s_star_index_node(3) = 's_star_3'
```

Special boxes: choose your title

(in the .rst file)

```
.. topic:: Tiny disclaimer:  
  
As I enjoy Project Euler greatly, I was hesitant to publish my solution online.  
  
Yet, since I saw many similar solutions are already available, I decided the educational  
cause of this tutorial justifies the means.  
  
So all you Euler buddies out there – hope you share my view :)
```



Tiny disclaimer:

As I enjoy Project Euler greatly, I was hesitant to publish my solution online.

Yet, since I saw many similar solutions are already available, I decided the educational cause of this tutorial justifies the means.

So all you Euler buddies out there - hope you share my view :)

(in the .rst
file)

Insert code

```
This page was generated using this .rst code:  
*****  
|  
|  
.. literalinclude:: input_parser.rst
```



This page was generated using this .rst code:

```
Input Parser  
=====|  
|  
.. automodule:: src.input_parser  
:members: input_parser  
|
```

```
This page was generated using this .rst code:  
*****
```

```
|  
.. literalinclude:: input_parser.rst  
|
```

```
:ref:`Return Home <mastertoc>`
```

Special formatting: textcolor

```
.. role:: pinktext

.. To make sure this is indeed colored pink, I added a .pinktext class to graphite.css

:pinktext:`Hope you find this useful!`
```

(in the .rst
file)

```
.pinktext {  
    color: #d481e6;  
}
```

(in
graphite.css)

Hope you find this useful!

Special formatting: breakline and code

(in the .rst
file)

```
.. |br| raw:: html

<br />

#. ``pip install sphinx``

#. Copy the ``documentation_template_for_your_next_project`` folder in this repository, |br|
   make it a subdirectory in your local project and rename it ``documentation``.
```



1. pip install sphinx

2. Copy the documentation_template_for_your_next_project folder in this repository,
make it a subdirectory in your local project and rename it documentation.

Links: to a specific item

(in .rst file:
where the link
will be)

```
:ref:`Return Home <mastertoc>`
```

```
.. toctree::  
    :maxdepth: 1  
    :name: mastertoc
```

(in .rst file:
where the link
directs)

[Return Home](#)

Links: to a page in the project

(in .rst file)

```
To use this for YOUR next project, browse this documentation and follow the instructions  
:doc:`here <how_to_use_this_for_your_next_project>`.
```



To use this for YOUR next project, browse this documentation and follow the instructions [here](#).

Links: to the web

(in .rst file)

The goal of this project is to help make `Sphinx <<http://www.sphinx-doc.org/en/master/>>`_ accessible and easy to use.



The goal of this project is to help make Sphinx accessible and easy to use.

THE
END

(for reals this time)