Pygorithm module in Python

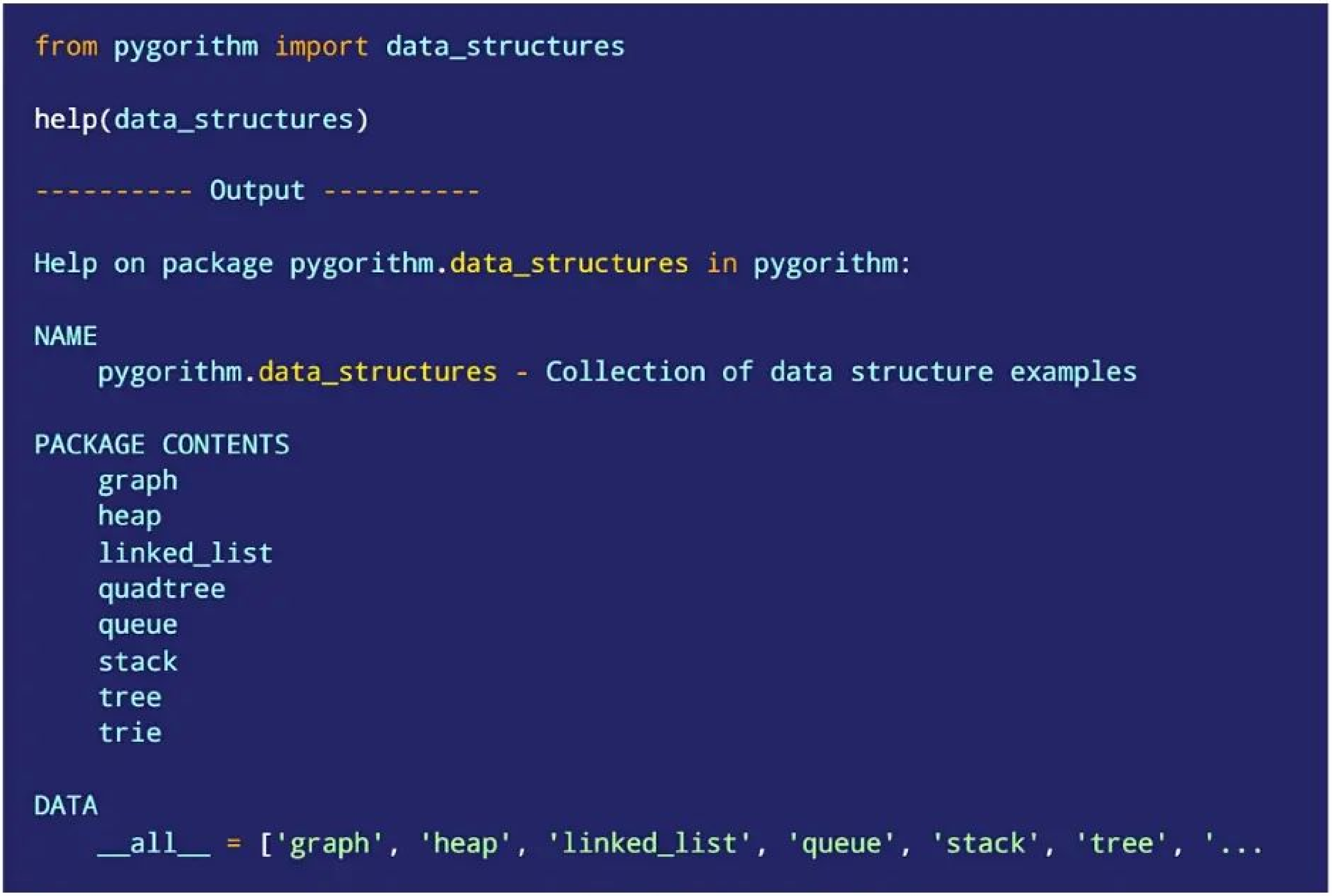
Pygorithm module is written in pure python and for the educational purpose. It helps to learn all the major algorithms on the go.

This module provides code implementation of various algorithms in Python.

To use this module first we need to install it using below pip command.

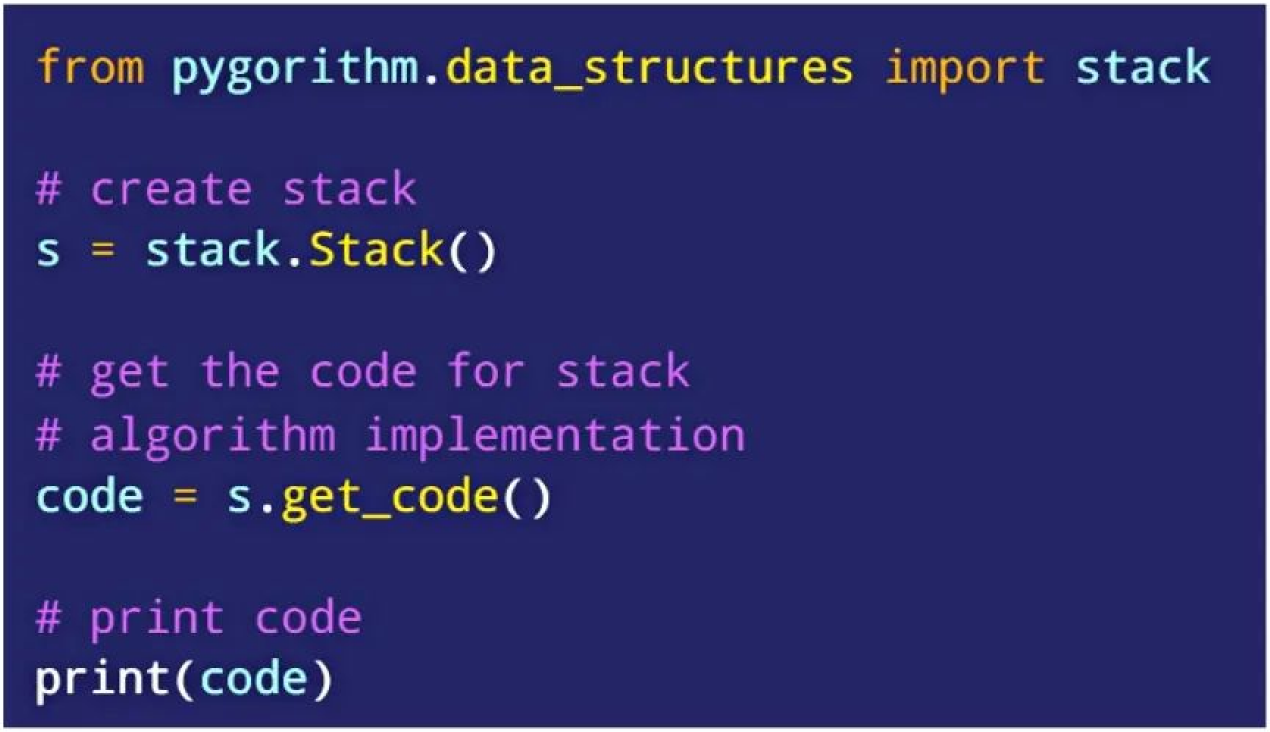
```
pip install pygorithm
```

Find all the available data structures:



```
from pygorithm import data_structures

help(data_structures)

---------- Output ----------

Help on package pygorithm.data_structures in pygorithm:

NAME
    pygorithm.data_structures - Collection of data structure examples

PACKAGE CONTENTS
    graph
    heap
    linked_list
    quadtree
    queue
    stack
    tree
    trie

DATA
    __all__ = ['graph', 'heap', 'linked_list', 'queue', 'stack', 'tree', '...
```

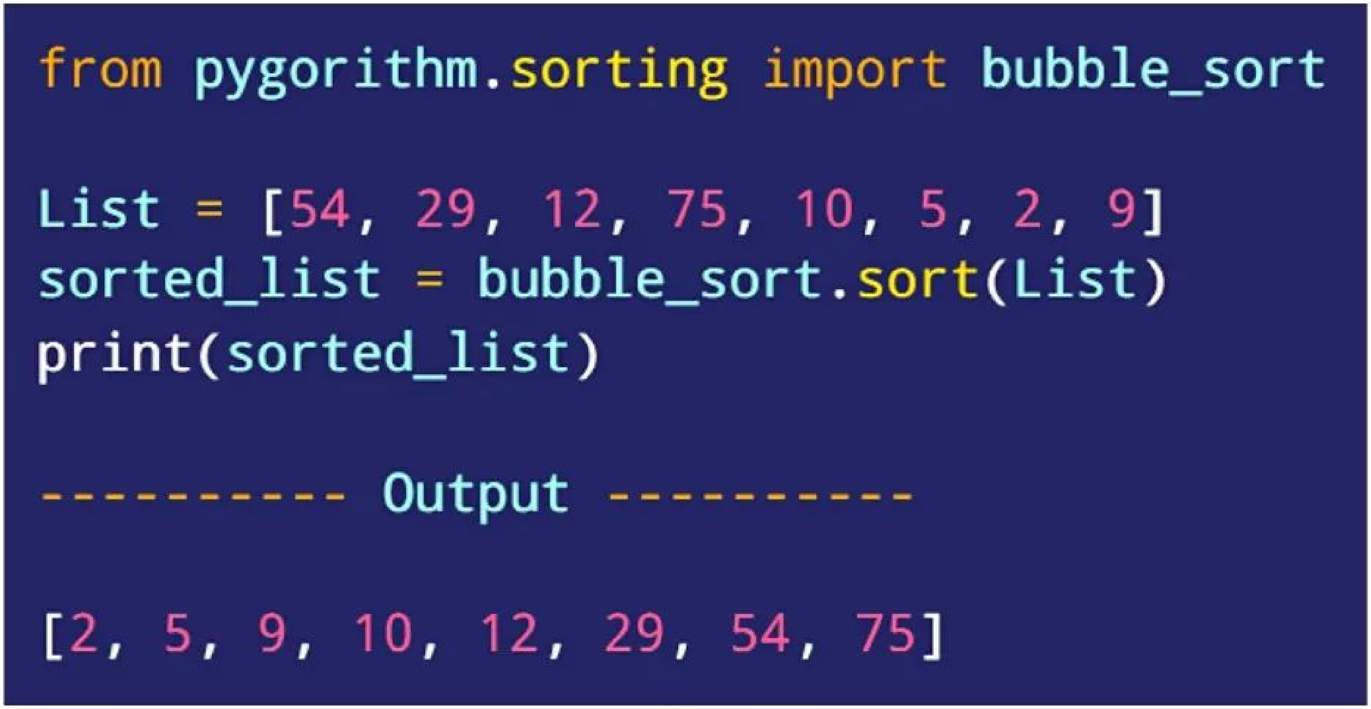
Get the implementation code of stack data_structure:



```python
from pygorithm.data_structures import stack

# create stack
s = stack.Stack()

# get the code for stack
# algorithm implementation
code = s.get_code()

# print code
print(code)
```

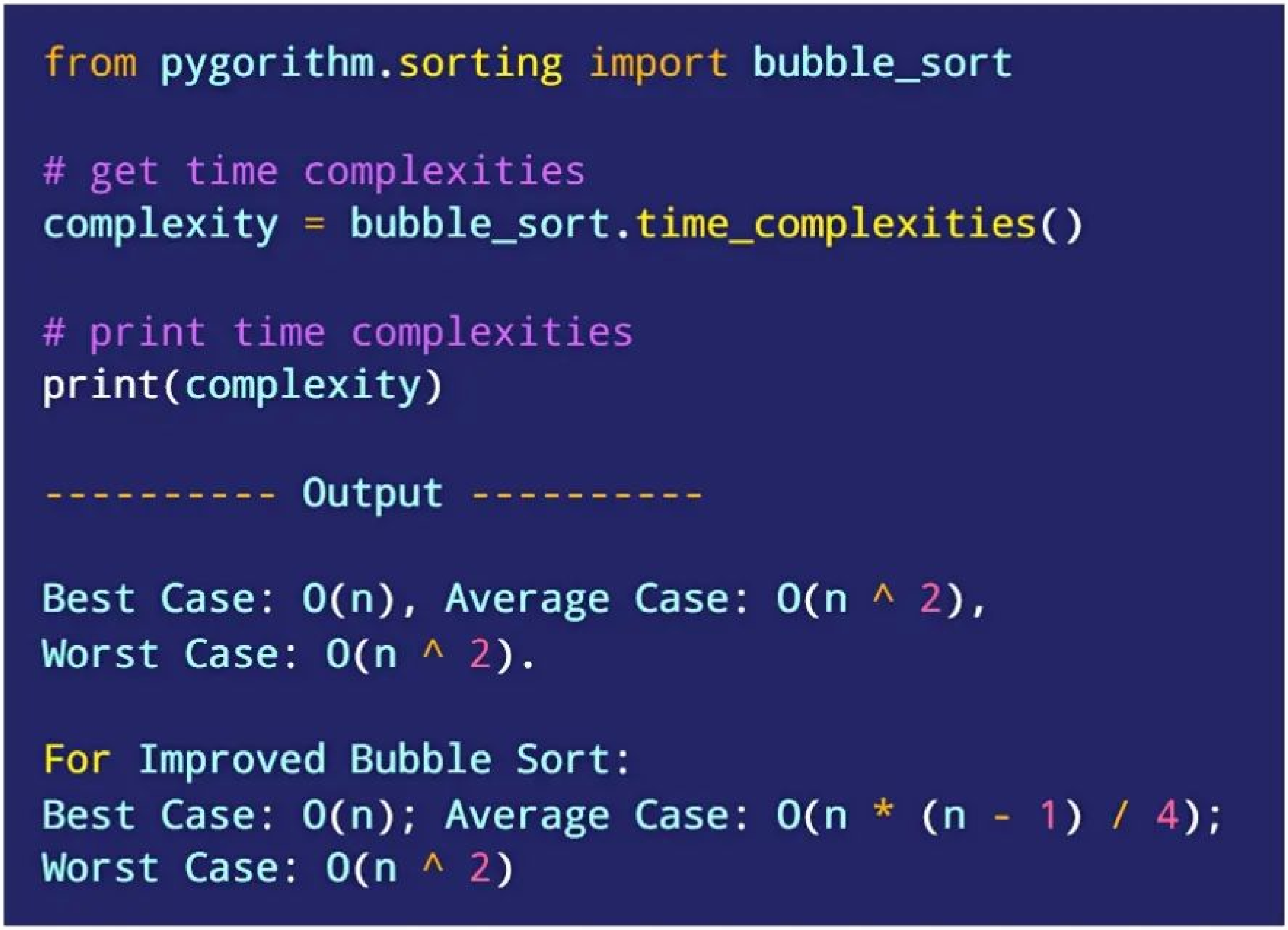
This codew will return the complete code for stack implementation.

Apply bubble sort to the list:



```
from pygorithm.sorting import bubble_sort

List = [54, 29, 12, 75, 10, 5, 2, 9]
sorted_list = bubble_sort.sort(List)
print(sorted_list)

---------- Output ----------

[2, 5, 9, 10, 12, 29, 54, 75]
```

Get the time complexities:



```
from pygorithm.sorting import bubble_sort

# get time complexities
complexity = bubble_sort.time_complexities()

# print time complexities
print(complexity)

---------- Output ----------

Best Case: O(n), Average Case: O(n ^ 2),
Worst Case: O(n ^ 2).

For Improved Bubble Sort:
Best Case: O(n); Average Case: O(n * (n - 1) / 4);
Worst Case: O(n ^ 2)
```

Save this post if you found it helpful.

THANK YOU!