Issues Found

Installation.. 2
Coding.. 3

Installation

Conflicts updating spyder

Collecting package metadata (current_repodata.json): done

Solving environment: failed with initial frozen solve. Retrying with flexible solve.

Solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.

Collecting package metadata (repodata.json): done

Solving environment: failed with initial frozen solve. Retrying with flexible solve.

Solving environment: /

Found conflicts! Looking for incompatible packages.

This can take several minutes. Press CTRL-C to abort.

Solution

conda update anaconda

conda update spyder

conda install spyder=4.0.1

Coding

```
# Strategy:  Iterate over a copy
for user, status in users.copy().items():
    if status == 'inactive':
        del users[user]
```

Error

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call
last)
<ipython-input-219-9bd61c12fc06> in <module>
      3
      4 # Strategy:  Iterate over a copy
----> 5 for user, status in users.copy().items():
      6     if status == 'inactive':
      7         del users[user]

NameError: name 'users' is not defined


---------------------------------------------------------------------------
```

Solution:

Create users list

users = {'user': 'Brown Rogers', 'status': 'inactive', 'user1': 'Adam Rogers', 'status1': 'active'}

```
# Code that modifies a collection while iterating over that same collection can be tricky to get right.
# Instead, it is usually more straight-forward to loop over a copy of the collection or to create a new collection

# Strategy:  Iterate over a copy

users = {'user': 'Brown Rogers', 'status': 'inactive', 'user1': 'Adam Rogers', 'status1': 'active'}

for user, status in users.copy().items():
    if status == 'inactive':
        del users[user]

print(users.items())
```

```
dict_items([('user', 'Brown Rogers'), ('user1', 'Adam Rogers'), ('status1', 'active')])
```

```
# Strategy:  Create a new collection
active_users = {}
for user, status in users.items():
    if status == 'active':
        active_users[user] = status
```

```
{'status1': 'active'}
```

Code changed to print T when the user is active

```
# Strategy:  Create a new collection
active_users = {'user': 'Brown Rogers', 'status': 'inactive', 'user1': 'Adam Rogers', 'status1': 'active'}
for user, status in users.items():
    if status == 'active':
        active_users[user] = 'T'

print(active_users)
```

```
{'user': 'Brown Rogers', 'status': 'inactive', 'user1': 'Adam Rogers', 'status1': 'T'}
```

Error

```
# In a function call, keyword arguments must follow positional arguments.
# All the keyword arguments passed must match one of the arguments accepted by the function

def function(a):
    pass

function(0, a=0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-3-ef0834e9ff95> in <module>
      5     pass
      6
----> 7 function(0, a=0)

TypeError: function() got multiple values for argument 'a'
```

Solution

Call function a with one value 0 or a = 0

Error

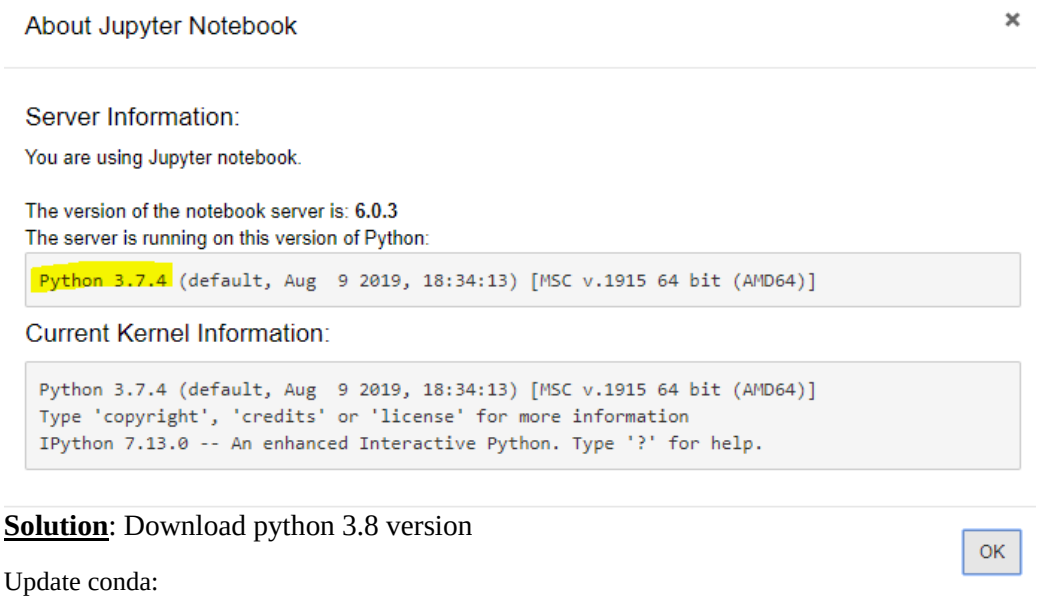
In Python 3.8, you can use / to denote that all arguments before it must be specified by position. You can rewrite incr() to only accept positional arguments

```
# pos_only_arg is restricted to only use positional parameters as there is a /
def pos_only_arg(arg, /):
    print(arg)

pos_only_arg(1)
```

```
  File "<ipython-input-113-1df7609c3c9b>", line 2
    def pos_only_arg(arg, /):
                          ^
SyntaxError: invalid syntax
```

Version running



Solution: Download python 3.8 version

Update conda:

conda update -n base -c defaults conda

Create new environment with Python 3.8:

conda create -n python38 python=3.8

Activate your new Python 3.8 environment:

conda activate python38

Start Python 3.8 with any of these commands:

python

jupyter notebook

ipython

```
Python 3.8.1 (default, Mar  2 2020, 13:06:26) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> def pos_only_arg(arg, /):
...     print(arg)
...
>>> pos_only_arg(1)
1
```

Error

```
# last uses all three calling conventions in the same function definition
def combined_example(pos_only, /, standard, *, kwd_only):
    print(pos_only, standard, kwd_only)

combined_example(1, 2, kwd_only=3)
combined_example(1, standard=2, kwd_only=3)
```

```
  File "<ipython-input-2-854084fe04c0>", line 2
    def combined_example(pos_only, /, standard, *, kwd_only):
                                      ^
SyntaxError: invalid syntax
```

Solution

```
>>> def combined_example(pos_only, /, standard, *, kwd_only):
...     print(pos_only, standard, kwd_only)
...
>>> combined_example(1, 2, kwd_only=3)
1 2 3
>>> combined_example(1, standard=2, kwd_only=3)
1 2 3
>>>
```

Error

```
# Right approach
def foo(name, /, **kwds):
    return 'name' in kwds

foo(1, **{'name': 2})

# The use case will determine which parameters to use in the function definition

def f(pos1, pos2, /, pos_or_kwd, *, kwd1, kwd2):
```

```
  File "<ipython-input-125-5499d06fbf8e>", line 2
    def foo(name, /, **kwds):
                   ^
SyntaxError: invalid syntax
```

Solution

```
>>> def foo(name, /, **kwds):
...     return 'name' in kwds
...
>>> foo(1, **{'name': 2})
True
```