Breaking Python

The Christie NHS
NHS Foundation Trust

You will encounter different kinds of errors:

Syntax errors: These occur when converting the source code into byte code. Something is wrong with the syntax – Missing semicolon or unclosed brackets etc… These will appear straight away when you run your code

Runtime errors: These go wrong when the code is being run. Some of the code may work before it suddenly breaks… Python will print an exception which can be useful in debugging.

Semantic/Logical errors: The code doesn't do what you think it's doing. These are harder to spot because the code doesn't crash. It may appear to run fine it just might not give you the answer you expect. It does what you told it to do.

Type of error	What went wrong
SyntaxError	Typo. Missing quotes, colons or brackets. Used a keyword as a variable name. Used = instead of == in a conditional.
IndentationError	A mixture of tabs and spaces have been used. Not all lines in a block are indented equally.
AttributeError	Calling a method that doesn’t exist for a given object
TypeError	Wrong number of arguments for a function. Tried to use an operator on the wrong type e.g. add an integer to a string. An object is None instead of a value.
NameError	Tried to use a variable that doesn’t exist. Local variables cannot be referred to from outside the function in which they are defined.

Type of error	What went wrong
IOError	Tried to open a file that doesn’t exist.
KeyError	Tried to access a dictionary with a key value that doesn’t exist in the dictionary
IndexError	The index used to access a list, string, or tuple is greater than its length minus one.
UnboundLocalError	Tried to reference a variable before it was assigned.
ValueError	Tried to perform an operation on a value that doesn’t exist. Tried to convert a value that is not valid e.g. convert a string to an int.

Some examples…

The Christie NHS
NHS Foundation Trust

Name Error

Broken Code

Fixed Code

```
1  a = 1
2  b = 2
3  c = 3
4  print d
5
```

```
Traceback (most recent call last):
  File ".\BadCode.py", line 4, in
<module>
    print d
NameError: name 'd' is not defined
```

Index Error

Broken Code

Fixed Code

```
1  a = [1,2,3]
2  print a[3]
3
4
5
```

```
Traceback (most recent call last):
  File ".\BadCode.py", line 2, in
<module>
    print a[3]
IndexError: list index out of range
```

Indentation Error

Broken Code

Fixed Code

```
1  for i in range(0,5):
2      print i
3        print i+1
4
5
```

```
  File ".\BadCode.py", line 3
    print i+1
    ^
IndentationError: unexpected indent
```

Attribute Error

Broken Code

Fixed Code

```
1 a = [1,2,3,4]
2 a.flip()
3
4 print a
5
```

```
Traceback (most recent call last):
  File ".\BadCode.py", line 2, in <module>
    a.flip()
AttributeError: 'list' object has no
attribute 'flip'
```

Type Error

Broken Code

Fixed Code

```
1 a = “Number: “
2 b = 7
3
4 print a+b
5
```

```
Traceback (most recent call last):
  File ".\BadCode.py", line 4, in <module>
    print a+b
TypeError: cannot concatenate 'str' and
'int' objects
```

The Christie NHS
NHS Foundation Trust

Type Error

Broken Code

Fixed Code

```
1 def
2 myfunction(a,b):
3       print a,b
4
5 myfunction(1,2,3)
```

```
Traceback (most recent call last):
  File ".\BadCode.py", line 4, in <module>
    myfunction(1,2,3)
TypeError: myfunction() takes exactly 2
arguments (3 given)
```

IO Error

Broken Code

Fixed Code

```
with open("a.txt") as a:
    for line in a:
        print line


```

```
 File ".\BadCode.py", line 1, in <module>
    with open("a.txt") as a:
IOError: [Errno 2] No such file or
directory: 'a.txt'
```

The Christie NHS
NHS Foundation Trust

Syntax Error

Broken Code

Fixed Code

```
1 a = [1,2,3]
2
3 for i in a
4     print i
5
```

```
 File ".\BadCode.py", line 3
    for i in a
             ^
SyntaxError: invalid syntax
```

Syntax Error

Broken Code

Fixed Code

```
1 from = “a”
2 to = “b”
3
4 print from, to
5
```

```
  File ".\BadCode.py", line 1
    from = "a"
         ^
SyntaxError: invalid syntax
```

Syntax Error

Broken Code

Fixed Code

```
1 i = 4
2 if i=4:
3     print “ok”
4
5
```

```
  File ".\BadCode.py", line 2
    if i=4:
        ^
SyntaxError: invalid syntax
```

Some things already have definitions and you can't change them. These are keywords:

and	as	assert	break
class	continue	def	del
elif	else	except	exec
finally	for	from	global
if	import	in	is
lambda	not	or	pass
print	raise	return	try
while	with	yield	

The Christie NHS
NHS Foundation Trust

When your code contains functions it may appear to break in more than one place….

```
Traceback (most recent call last):
  File "C:/Users/Matthew/Documents/Python
Scripts/colourmaps.py", line 50, in <module>
    plot_colour_gradients(cmap_category, cmap_list)
  File "C:/Users/Matthew/Documents/Python
Scripts/colourmaps.py", line 34, in
plot_colour_gradients
    fig, axes = plt.subplots(a, nrows=nrows)
NameError: global name 'a' is not defined
```

Here the function `plot_colour_gradients` is called on line 50 but the actual issue is on line 34 where a Name Error occurs. Read the traceback from bottom to top. Most of the useful information is at the bottom.

The Christie NHS
NHS Foundation Trust

Semantic/Logical errors…

Semantic Error

Broken Code

```
a = 5
b = 2
mean = a+b/2
print mean

```

```
>> 6
```

Fixed Code

Semantic Error

Broken Code

```
a = [1,2,3,4,5]
a = a.reverse()

print a

```

```
>> None
```

Fixed Code

Semantic Error

Broken Code

```
# print 1 to 10
for i in range(1,10):
    print i,
```

```
>> 1 2 3 4 5 6 7 8 9
```

Fixed Code

Use print statements to help to diagnose semantic errors.
Make your print statements more descriptive so that you know what you're looking at.

Instead of:

```
print a
```

Use:

```
print "a = {}".format(a)
```

The Christie NHS
NHS Foundation Trust

Run your code every time you add a few lines of code to check that they work as expected. This will help to identify where things are going wrong.

If things aren’t working you can comment them out but please don’t just delete it before asking for help!

More on errors here:

https://docs.python.org/3/library/exceptions.html