Python Errors Guide

Getting errors in your code at some point is as expected as the sky being blue tomorrow. Everyone gets errors in their code, and it's important to understand how to read them and address them. This list is not exhaustive, but serves as a guide on how to deal with the most common errors you will come across in your respective Python journeys

NameError

A **NameError** occurs when you try to use a variable or function in a way it shouldn't be. Some things that can cause an error are:

- **Using a variable or function name that has yet to be defined**

```
print(school)
school = "Howard University"
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
Traceback (most recent call last):
  File "/home/codio/workspace/notes.py", line 1, in <module>
    print(school)
          ^^^^^^
NameError: name 'school' is not defined
```

- **Misspelling a variable name or function**

```
num_students = 192
print(numstudents)
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
Traceback (most recent call last):
  File "/home/codio/workspace/notes.py", line 2, in <module>
    print(numstudents)
          ^^^^^^^^^^^
NameError: name 'numstudents' is not defined. Did you mean:
'num_students'?
```

SyntaxError

SyntaxErrors happen when the Python interpreter doesn't understand the structure of the code. Ways that this usually happens are:

- **Misplaced, missing, or mismatched punctuation (parentheses, square brackets, etc)**

```
num = int(input("Enter a number")
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
  File "/home/codio/workspace/notes.py", line 1
    num = int(input("Enter a number")
              ^
SyntaxError: '(' was never closed
```

- **Misspelled or missing python keywords**

```
year = int(input("Enter the year"))
if year < 2000:
 print("earlier than 2000")
elf year == 2000:
 print("currently 2000")
else:
 print("later than 2000")
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
  File "/home/codio/workspace/notes.py", line 4
    elf year == 2000:
        ^^^^
SyntaxError: invalid syntax
```

- **Illegal characters in variable names**

```
hello!world = "Hello world!"
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
  File "/home/codio/workspace/notes.py", line 1
    hello!world = "Hello world!"
         ^
SyntaxError: invalid syntax
```

- **Incorrect indentation (this will actually show an IndentationError, which is a kind of syntax error)**

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
  File "/home/codio/workspace/notes.py", line 2
    print(elem)
    ^
IndentationError: expected an indented block after 'for' statement on
line 1
```

- **Incorrect use of the assignment operator (=)**

```
if "sza" = "SZA":
 print("yes")
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
  File "/home/codio/workspace/notes.py", line 1
    if "sza" = "SZA":
       ^^^^^
SyntaxError: cannot assign to literal here. Maybe you meant '=='
instead of '='?
```

TypeError

TypeErrors happen when the datatype of an object or variable is not compatible with an operation. Some things that can cause these kinds of errors are:

- **Performing an operation on two incompatible data types (i.e. adding a string and an integer, doing comparisons on a string and integer, etc)**

```
year = 2024
print("The year is " + year)
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
Traceback (most recent call last):
  File "/home/codio/workspace/notes.py", line 2, in <module>
    print("The year is " + year)
          ~~~~~~~~~~~~~~~~^~~~~~
TypeError: can only concatenate str (not "int") to str
```

- **Iterating over a non-iterative value**

```
for num in 25:
    print(num)
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
Traceback (most recent call last):
  File "/home/codio/workspace/notes.py", line 1, in <module>
    for num in 25:
TypeError: 'int' object is not iterable
```

- **Passing an incorrect type to a built-in function**

```
print(len(5))
```

```
codio@nissanmission-finlandperson:~/workspace$ python3 notes.py
Traceback (most recent call last):
  File "/home/codio/workspace/notes.py", line 1, in <module>
    print(len(5))
          ^^^^^^
TypeError: object of type 'int' has no len()
```

How to fix these errors

Fixing errors when you get them is pretty easy! There are just a couple steps you need to take:

1. **Figure out where the error is**
    a. As shown above, error messages will always tell you what file and line the error is located. Take the time to read the error message carefully
    b. Also take a look at the line above and below the line the error message mentions. So, if the error message says the issue is on line 2, take a look at lines 1 and 3 as well
2. **Keep a lookout for hints the error message gives.**
    a. As Python has progressed over the years, the quality of error messages has progressed as well. Error messages now will sometimes have hints as to how to fix the error, even underlining exactly where the issue is. Try doing what the hint says and see if it works
3. **Take a good look at your code**
    a. Once you've read the error message and have a general idea where the error might be, take a good look at your code and check for any misspellings, missing colons, etc.