Debugging Errors: Takeaways ⤴

Syntax

- SyntaxError occurs when the Python interpreter can't parse the code as valid Python code:
    - Input:
      ```
      the_answer = "42
      ```
    - Output:
      ```
      File "ipython-input-1-85ffad3b5465", line 1
          the_answer = "42
                         ^
      SyntaxError: EOL while scanning string literal
      ```
- IndentError occurs when the Python interpreter can't accurately determine a line's intended indentation:
    - Input:
      ```
      def find():
          print("42")
           print("what, really?")
      ```
    - Output:
      ```
      File "ipython-input-2-dd6a6ca22a8f", line 3
          print("what, really?")
          ^
      IndentationError: unexpected indent
      ```

- TypeError occurs when the Python interpreter didn't know how to apply a function or operator to a specific value type:
  - Input:

```
42 + "42"
```

  - Output:

```
TypeError
ipython-input-5-6061d7365225 in ()

    42 + "42"

TypeError unsupported operand type(s) for +: 'int' and 'str'
```

- ValueError occurs when an operation or function received a value of the right type but an inappropriate value:
  - Input:

```
float("guardians")
```

  - Output:

```
ValueError
ipython-input-8-325ad4dda32a in ()

    float("guardians")

ValueError could not convert string to float: 'guardians'
```

- IndexError:
  - Input:
    ```
    lives= [1,2,3]

    lives[4]
    ```
  - Output:
    ```
    IndexError                          Traceback(most recent call last)

    ipython-input-9-e6921ecc6eca in ()

          1 lives= [1,2,3]

    ----> 2 lives[4]

    IndexError: list index out of range
    ```

- AttributeError:
  - Input:
    ```
    f = open("story.txt")

    f.split("")
    ```
  - Output:
    ```
    AttributeError                      Traceback(most recent call last)

    ipython-input-10-4cad13e731ca in ()

          1 f = open("story.txt")

    ----> 2 f.split("")

    AttributeError: '_io.TextIOWrapper' object has no attribute 'split'
    ```

Concepts

- The two main types of errors are:
  - Syntax errors. Some examples of syntax errors include:
    - Missing ending quotes or starting quotes
    - Using improper indentation
    - Using improper keywords
  - Runtime errors. Some examples of runtime errors include:
    - Calling a function before it's defined
    - Calling a method or attribute that the object doesn't contain
    - Attempting to convert a value to an incompatible data type

Resources

- Python Documentation: Errors and Exceptions

- Python Documentation: Errors and Exceptions