Errors

Python errors, also known as exceptions. Errors in Python are events that occur during the execution of a program that disrupts the normal flow of the program's instructions. When an error occurs, Python generates an exception, which is a special kind of object that can be handled to prevent the program from crashing.

Here are some common types of errors in Python along with examples:

1. **SyntaxError:**
   This error occurs when the Python interpreter encounters a syntax that is not valid Python code.

   Example:
   ```python
   # SyntaxError: invalid syntax
   print "Hello, World!"
   ```

2. **IndentationError:**
   Python relies on indentation to define blocks of code. An IndentationError occurs if there is an issue with the indentation.

   Example:
   ```python
   # IndentationError: unexpected indent
   if True:
   print("Indentation is important!")
   ```

3. **NameError:**
   This error occurs when a name is not found in the local or global scope.

   Example:
   ```python
   # NameError: name 'variable' is not defined
   print(variable)
   ```

4. **TypeError:**
   This error occurs when an operation or function is applied to an object of the wrong type.

   Example:
   ```python
   # TypeError: can't multiply sequence by non-int of type 'str'
   result = "Hello" * "World"
   ```

5. **ValueError:**

This error occurs when a built-in operation or function receives an argument of the correct type but an invalid value.

Example:
```python
# ValueError: invalid literal for int() with base 10: 'abc'
number = int('abc')
```

6. **IndexError:**
This error occurs when trying to access an index that does not exist in a sequence (e.g., list, tuple).

Example:
```python
# IndexError: list index out of range
my_list = [1, 2, 3]
print(my_list[4])
```

7. **FileNotFoundError:**
This error occurs when trying to open or access a file that does not exist.

Example:
```python
# FileNotFoundError: [Errno 2] No such file or directory: 'nonexistent.txt'
with open('nonexistent.txt', 'r') as file:
    content = file.read()
```

8. **ZeroDivisionError:**
This error occurs when trying to divide a number by zero.

Example:
```python
# ZeroDivisionError: division by zero
result = 10 / 0
```

Handling exceptions is crucial to creating robust and error-tolerant Python programs. This is typically done using `try`, `except`, `else`, and `finally` blocks. For example:

```python
try:
    result = 10 / 0
except ZeroDivisionError as e:
    print(f"Error: {e}")
else:
    print("No error occurred.")
finally:
```

```
    print("This will execute no matter what.")
```

This code will catch the `ZeroDivisionError` and print an error message, then proceed to the `finally` block.