# Run Time Errors

In Python, runtime errors occur during the execution of a program. These errors are not detected by the Python interpreter until the program is running. They are often the result of logical issues or unexpected conditions that arise while the program is being executed. Here are some common runtime errors in Python with examples:

1. **ZeroDivisionError:**
   This error occurs when attempting to divide a number by zero.

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

2. **IndexError:**
   Raised when trying to access an index that is outside the range of valid indices for a sequence (e.g., list, tuple).

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

3. **TypeError:**
   This error occurs when an operation is performed on an object of an inappropriate type.

   ```python
   # TypeError: unsupported operand type(s) for +: 'int' and 'str'
   result = 10 + "5"
   ```

4. **ValueError:**
   Raised when a built-in operation or function receives an argument of the correct type but with an invalid value.

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

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

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

```
    content = file.read()
```

6. **AttributeError:**
   Raised when trying to access an attribute or method that does not exist for a given object.

   ```python
   # AttributeError: 'int' object has no attribute 'length'
   length = (10).length
   ```

7. **KeyError:**
   Raised when trying to access a dictionary key that does not exist.

   ```python
   # KeyError: 'nonexistent_key'
   my_dict = {'key1': 'value1'}
   value = my_dict['nonexistent_key']
   ```

8. **ImportError:**
   Raised when an import statement fails to import a module or name.

   ```python
   # ImportError: No module named 'nonexistent_module'
   import nonexistent_module
   ```

Handling runtime errors is important for creating robust and user-friendly programs. This is often done using `try`, `except`, `else`, and `finally` blocks to gracefully handle exceptions and prevent the program from crashing.