In [2]:

```
class DataFormatError(Exception):
    """Raised when data isn't in the expected format."""
    pass

class EmptyFileError(Exception):
    """Raised when the file is empty."""
    pass


def read_and_process_file(file_path):
    try:
        with open(file_path, 'r') as file:
            data = file.read().strip()


            if not data:
                raise EmptyFileError("The file is empty.")


            numbers = [int(num) for num in data.split()]


            if len(numbers) < 2:
                raise DataFormatError("Data must contain at least two numbers.")


            result = sum(numbers) / len(numbers)
            return result

    except FileNotFoundError:
        print("Error: The file was not found. Please check the file path.")
    except ValueError:
        print("Error: The file contains non-integer values that cannot be processed.")
    except TypeError:
        print("Error: There was a type error in the data processing.")
    except DataFormatError as e:
        print(f"DataFormatError: {e}")
    except EmptyFileError as e:
        print(f"EmptyFileError: {e}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")


file_path = "sample_data.txt"
result = read_and_process_file(file_path)
if result:
    print(f"The average of the numbers is: {result}")
```

```
EmptyFileError: The file is empty.
```

In [4]:

```
class DataFormatError(Exception):
    """Raised when data isn't in the expected format."""
    pass

class EmptyFileError(Exception):
    """Raised when the file is empty."""
```

```
    pass


def read_and_process_file(file_path):
    try:
        with open(file_path, 'r') as file:
            data = file.read().strip()


            if not data:
                raise EmptyFileError("The file is empty.")


            numbers = [int(num) for num in data.split()]


            if len(numbers) < 2:
                raise DataFormatError("Data must contain at least two numbers.")


            result = sum(numbers) / len(numbers)
            return result

    except FileNotFoundError:
        print("Error: The file was not found. Please check the file path.")
    except ValueError:
        print("Error: The file contains non-integer values that cannot be processed.")
    except TypeError:
        print("Error: There was a type error in the data processing.")
    except DataFormatError as e:
        print(f"DataFormatError: {e}")
    except EmptyFileError as e:
        print(f"EmptyFileError: {e}")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")


file_path = "ample_data.txt"
result = read_and_process_file(file_path)
if result:
    print(f"The average of the numbers is: {result}")
```

```
Error: The file contains non-integer values that cannot be processed.
```

In [ ]: