Exceptions - MAD 102 Week 12 Notes

Hia Al Saleh

November 20th, 2024

Contents

1 Handling Errors **2**

2 Exception Handling **2**

3 Try / Except **2**

4 Try / Except Example **2**

5 Exception Errors **2**

6 Multiple Exception Handlers **3**

7 Raising Exceptions **3**

8 Exceptions with Functions **3**

9 `finally` Clause **3**

10 Creating Custom Error Types **3**

1 Handling Errors

- When the user enters incorrect information – how do you handle it?
- How do you detect that the information is incorrect and what course of action do you take to ensure that your program continues to run?
- Create error-checking code to detect and handle errors while the program is executing.

2 Exception Handling

- Special constructs known as **exception-handling** handle exceptional conditions.
- Example: User inputs `'dog'` instead of a number.

3 Try / Except

- If there's a possibility that your code may produce an exception, place the code in a `try` block.
- Use the keyword `try` to mark the block.
- The solution for handling the exception is placed in an `except` block.
- Use the keyword `except` to mark the block.

4 Try / Except Example

- If no errors occur in the `try` block (no exceptions), the code progresses as normal.
- If an error occurs (an exception is thrown), the code in the `except` block is executed.
- Any code in the `try` block is skipped if an error occurs.

5 Exception Errors

- **ValueError** – Raised when an operation or function receives an argument with the right type but inappropriate value.
- **KeyError** – Raised when a mapping (dictionary) key is not found in the set of existing keys.
- **IndexError** – Raised when a sequence subscript is out of range.

- **NameError** – Raised when an identifier is not found.
- **ZeroDivisionError** – Raised when the second argument of a division or modulo operation is zero.

More built-in exception errors can be found in the Python documentation: `https://docs.python.org/3/library/exceptions.html`.

6 Multiple Exception Handlers

- Multiple exceptions could be generated depending on your program.
- Use multiple `except` blocks to handle various types of exceptions.
- Each `except` block should specify the type of exception it handles.

Example:

```
except (ValueError, TypeError):
    # Code to execute
```

7 Raising Exceptions

- Exception handling constructs can be used to raise (throw) errors.
- The keyword `raise` is used, followed by the type of error to raise.
- A string argument is provided to the exception error that details the issue.

8 Exceptions with Functions

- Exceptions can be part of functions, allowing for better program clarity.

9 `finally` Clause

- The `finally` clause of a `try` statement specifies clean-up actions.
- These actions are always executed, regardless of whether an exception is raised.

10 Creating Custom Error Types

- Custom exception types can be created using classes.
- Define a class with an initializer and use it as a normal exception.