Lisbon School of Economics & Management
Universidade de Lisboa

LISBOA
UNIVERSIDADE DE LISBOA

Carlos J. Costa

HANDLING EXCEPTIONS IN PYTHON

Handling exceptions

- is a crucial aspect of writing robust and error-tolerant code.
- try and except blocks to catch and handle exceptions when they occur.

Handling a ZeroDivisionError

- Since dividing by zero is not allowed, a ZeroDivisionError will be raised.
- catch this exception with the except ZeroDivisionError block and print an error message.

```
try:
    numerator = 10
    denominator = 0
    result = numerator / denominator
except ZeroDivisionError:
    print("Error: Division by zero is not allowed.")
```

Handling a ValueError

- take user input and attempt to convert it to an integer using int().
- If the user enters something that is not a valid integer, a ValueError will be raised, and we handle it by printing an error message.

```
try:
    user_input = input("Enter an integer: ")
    number = int(user_input)
except ValueError:
    print("Error: Please enter a valid integer.")
```

Handling Multiple Exceptions

- In this example, we attempt to convert a non-integer string to an integer and perform a division by zero.

```
try:
    value = int("not an integer")
    result = 10 / 0
except ValueError:
    print("Error: Value conversion failed.")
except ZeroDivisionError:
    print("Error: Division by zero.")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

Using else and finally Blocks

- Take user input and perform a division.
- If no exceptions occur, the else block prints the result.
- The finally block always runs, whether an exception occurred or not, to indicate the completion of execution.

```
try:
    number = int(input("Enter a number: "))
    result = 10 / number
except ValueError:
    print("Error: Please enter a valid integer.")
except ZeroDivisionError:
    print("Error: Division by zero is not allowed.")
else:
    print(f"The result is: {result}")
finally:
    print("Execution completed.")
```

Handling exceptions in Python

```
try:
    # Code that may raise an exception
    num1 = int(input("Enter a numerator: "))
    num2 = int(input("Enter a denominator: "))
    result = num1 / num2
except ZeroDivisionError:
    # Handle the ZeroDivisionError exception
    print("Division by zero is not allowed.")
except ValueError:
    # Handle the ValueError exception (e.g., if the user enters a non-integer)
    print("Please enter valid integer values.")
except Exception as e:
    # Handle any other exceptions not explicitly caught above
    print(f"An error occurred: {e}")
else:
    # This block will run if no exceptions were raised
    print(f"The result of the division is: {result}")
finally:
    # This block always runs, whether an exception occurred or not
    print("Execution completed.")
```