19. How Do You Handle Exceptions With Try/Except/Finally In Python? Explain with coding snippets.

**Try Block:**

* **Purpose**: You use a try block to enclose code that might potentially raise an exception.
* **Execution**: Python attempts to execute the code inside the try block.
* **If No Exception**: If no exceptions occur, Python skips the except block(s) and moves on to any code following the try-except-finally structure.

**Except Block:**

* **Purpose**: An except block is used to specify how to handle specific types of exceptions that might occur within the preceding try block.
* **Usage**: You can have multiple except blocks to handle different types of exceptions separately. If an exception matches the type specified in an except block, Python executes the corresponding code.

**Finally Block:**

* **Purpose**: The finally block is optional and follows the try and except blocks. It is used to define cleanup actions that should be executed regardless of whether an exception occurred or not.
* **Execution**: Code in the finally block always runs, even if there was an exception in the try block or if you used a return, break, or continue statement to exit early from the try block.

try:

x = int(input("Enter a number: "))

result = 10 / x

print("Result:", result)

except ValueError:

print("Please enter a valid integer.")

except ZeroDivisionError:

print("Cannot divide by zero. Please enter a non-zero number.")

finally:

print("Execution completed.")

print("Program continues...")