**1. What is the role of try and exception block?**

**The try and except blocks are used to handle exceptions in Python. The try block is where you place code that might raise an exception. If an exception occurs, the program immediately jumps to the except block, which is used to handle the error gracefully and prevents the program from crashing.**

**2. What is the syntax for a basic try-except block?**

**python**

**Copy**

**try:**

**# Code that might raise an exception**

**result = 10 / 0 # Example: division by zero**

**except Exception as e:**

**# Code to handle the exception**

**print(f"An error occurred: {e}")**

**3. What happens if an exception occurs inside a try block and there is no matching except block?**

**If an exception occurs inside the try block and there is no matching except block, Python will propagate the exception, causing the program to terminate with an error message. If the exception is not handled at all, the program stops execution.**

**Example:**

**python**

**Copy**

**try:**

**# This will raise a ZeroDivisionError**

**result = 10 / 0**

**except ValueError:**

**# This block will not catch the exception**

**print("ValueError occurred")**

**Output:**

**vbnet**

**Copy**

**ZeroDivisionError: division by zero**

**4. What is the difference between using a bare except block and specifying a specific exception type?**

* **Bare except: Catches all exceptions, including unexpected ones. It’s not recommended because it can hide errors that might be important to identify.**

**Example of a bare except block:**

**python**

**Copy**

**try:**

**x = 1 / 0**

**except:**

**print("An error occurred")**

* **Specific exception type: It is recommended to specify the exact exception type, so only the intended errors are caught, making the code more robust and readable.**

**Example of a specific exception:**

**python**

**Copy**

**try:**

**x = 1 / 0**

**except ZeroDivisionError:**

**print("Cannot divide by zero")**

**5. Can you have nested try-except blocks in Python? If yes, then give an example.**

**Yes, you can have nested try-except blocks in Python, where one try-except block is inside another.**

**Example:**

**python**

**Copy**

**try:**

**try:**

**x = 10 / 0**

**except ZeroDivisionError:**

**print("Handled division by zero inside inner try")**

**raise**

**except ZeroDivisionError:**

**print("Handled division by zero in outer try")**

**6. Can we use multiple exception blocks, if yes then give an example.**

**Yes, you can use multiple except blocks to handle different types of exceptions separately.**

**Example:**

**python**

**Copy**

**try:**

**value = int(input("Enter a number: "))**

**result = 10 / value**

**except ZeroDivisionError:**

**print("Error: Division by zero")**

**except ValueError:**

**print("Error: Invalid input")**

**except Exception as e:**

**print(f"An unexpected error occurred: {e}")**

**7. Write the reason due to which following errors are raised:**

* **EOFError: Raised when one of the built-in functions (input(), read(), etc.) hits the end of the file and expects more data.**
* **FloatingPointError: Raised when a floating-point operation fails, such as a floating-point division by zero.**
* **IndexError: Raised when trying to access an index that is out of the range of a list or string.**
* **MemoryError: Raised when an operation runs out of memory.**
* **OverflowError: Raised when a numerical operation results in a number that is too large to be represented.**
* **TabError: Raised when there is an inconsistent use of tabs and spaces for indentation in the code.**
* **ValueError: Raised when an operation or function receives an argument that has the right type but an inappropriate value.**

**8. Write code for the following given scenario and add try-exception block to it:**

**a. Program to divide two numbers:**

**python**

**Copy**

**try:**

**num1 = float(input("Enter numerator: "))**

**num2 = float(input("Enter denominator: "))**

**result = num1 / num2**

**print(f"Result: {result}")**

**except ZeroDivisionError:**

**print("Error: Cannot divide by zero.")**

**except ValueError:**

**print("Error: Invalid input, please enter numeric values.")**

**b. Program to convert a string to an integer:**

**python**

**Copy**

**try:**

**string\_value = input("Enter a number: ")**

**integer\_value = int(string\_value)**

**print(f"Converted integer: {integer\_value}")**

**except ValueError:**

**print("Error: Input is not a valid integer.")**

**c. Program to access an element in a list:**

**python**

**Copy**

**my\_list = [1, 2, 3, 4, 5]**

**try:**

**index = int(input("Enter index to access: "))**

**print(f"Element at index {index}: {my\_list[index]}")**

**except IndexError:**

**print("Error: Index out of range.")**

**except ValueError:**

**print("Error: Invalid index, please enter an integer.")**

**d. Program to handle a specific exception:**

**python**

**Copy**

**try:**

**x = 1 / 0**

**except ZeroDivisionError:**

**print("Error: Division by zero is not allowed.")**

**e. Program to handle any exception:**

**python**

**Copy**

**try:**

**num = int(input("Enter a number: "))**

**result = 10 / num**

**print(f"Result: {result}")**

**except Exception as e:**

**print(f"An error occurred: {e}")**