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

**Ans.1** The try and except blocks are used in Python for handling exceptions. The try block contains the code that might raise an exception, and the except block handles the exception if it occurs, allowing the program to gracefully continue execution even if an error occurs.

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

**Ans.2**

**try:**

# Code that might raise an exception

**except ExceptionType:**

# Code to handle the exception

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

**Ans.3** If an exception occurs inside a try block and there is no matching except block to handle that specific exception type, the program will terminate and an error message will be displayed indicating that an unhandled exception occurred.

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

**Ans.4** Using a bare except block (except:) without specifying an exception type catches all exceptions, including built-in exceptions and custom exceptions. This is not recommended because it makes it difficult to understand the actual cause of the exception. Specifying a specific exception type (except ExceptionType:) allows you to catch and handle only that particular type of exception, providing more control over error handling.

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

**Ans.5**

**try:**

# Outer try block

x = 10 / 0

**except** ZeroDivisionError:

**try:**

# Inner try block

y = 10 / 2

**except** ZeroDivisionError:

print("Inner division by zero")

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

Ans.6

**try:**

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

result = 10 / num

**except** ValueError:

print("Invalid input, please enter a valid number.")

**except** ZeroDivisionError:

print("Cannot divide by zero.")

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

**a. EOFError :-** Raised when the end of file is reached unexpectedly during input operations.

**b. FloatingPointError :-** Raised when a floating-point operation fails to execute correctly, such as division by zero in floating-point arithmetic.

**c. IndexError :-** Raised when an index used to access an element in a sequence (like a list or tuple) is out of range.

**d. MemoryError :-** Raised when an operation runs out of memory.

**e. OverflowError :-** Raised when an arithmetic operation exceeds the limits of the data type.

**f. TabError :-** Raised when inconsistent use of tabs and spaces in the indentation of code is detected.

**g. ValueError :-** Raised when a built-in operation or function receives an argument of the correct 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**

**Ans.8 (a)**  **try:**

numerator = float(input("Enter the numerator: "))

denominator = float(input("Enter the denominator: "))

result = numerator / denominator

**print**("Result:", result)

**except** ZeroDivisionError:

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

**except** ValueError:

**print**("Invalid input. Please enter valid numbers.")

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

**Ans.8(b)**

**try:**

string\_num = input("Enter a number as a string: ")

integer\_num = int(string\_num)

**print**("Converted integer:", integer\_num)

**except** ValueError:

**print**("Invalid input. Please enter a valid integer.")

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

**Ans.8(c)**

**try:**

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

index = int(input("Enter an index: "))

value = my\_list[index]

print("Value at index", index, ":", value)

**except** IndexError:

print("Index out of range.")

**except** ValueError:

print("Invalid input. Please enter a valid index.")

d. Program to handle a specific exception

**Ans.8(d)**

**try:**

x = 10 / 0

**except** ZeroDivisionError:

print("Cannot divide by zero.")

**e. Program to handle any exception**

**Ans.8(e)**

**try:**

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

result = 10 / value

print("Result:", result)

**except** Exception as e:

print("An error occurred:", e)