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

= Try and Except statement is used to handle these errors within our code in Python. The try block is used to check some code for errors i.e the code inside the try block will execute when there is no error in the program. Whereas the code inside the except block will execute whenever the program encounters some error in the preceding try block.

try:

    k = 5//0 # raises divide by zero exception.

    print(k)

# handles zerodivision exception

except ZeroDivisionError:

    print("Can't divide by zero")

finally:

    # this block is always executed

    # regardless of exception generation.

    print('This is always executed')

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

= try:

# There can be errors in this block

except <error type>:

# Do this to handle exception;

# executed if the try block throws an error

else:

# Do this if try block executes successfully without errors

finally:

# This block is always executed

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

= If an exception occurs which does not match the exception named in the except clause, it is passed on to outer try statements; if no handler is found, it is an unhandled exception and execution stops with an error message.

program will immediately jump to the corresponding catch block that matches the type of the thrown exception. The catch block is where the program can handle the exception by executing specific code to address the exceptional situation. If there is no catch block that matches the type of the thrown exception, the program will terminate and an error message will be displayed.

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

= By not specifiycing an exception type, you might also loose information about the error itself. A bare except: clause will catch SystemExit and KeyboardInterrupt exceptions, making it harder to interrupt a program and can disguise other problems.

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

= In a Python program, if there is another try-except construct either inside either a try block or inside its except block, it is known as a nested-try block. This is needed when different blocks like outer and inner may cause different errors. To handle them, we need nested try blocks.

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

= In a Python program, if there is another try-except construct either inside either a try block or inside its except block, it is known as a nested-try block. This is needed when different blocks like outer and inner may cause different errors. To handle them, we need nested try blocks.

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

a. EOFError :

= this syntax error occurs when Python detects an unfinished statement or block of code. This can happen for many reasons, but the most likely cause is missing punctuation or an incorrectly indented block.

b. FloatingPointError :

= Floating-point decimal values generally do not have an exact binary representation. For this reason, you may experience some loss of precision, and some floating-point operations may produce unexpected results.

c. IndexError ;

= This is usually because the index goes out of bounds by being too large. For example, if you have a list with three items and you try to access the fourth item, you will get an IndexError. when an item from a list is attempted to be accessed that is outside the index range of the list.

d. MemoryError ;

= a lack of available memory on the system, inefficient code that creates too many objects, or a data set that is too large to fit into memory.

e. OverflowError;

= It happens when you're working with numbers and the result of your calculation becomes too big (or too small) to be handled by Python.

f. TabError ;

= when there is a mix of tabs and spaces within the same block of code. It relies on consistent indentation to define the structure of code blocks, such as loops, conditionals, and functions. Python script mixes tabs and spaces for indentation within the same block of code.

g. ValueError;

= ValueError in Python is raised when a function receives an argument of the correct type but an inappropriate value. when a function is called with the proper argument type but with the wrong value. This mistake frequently occurs during mathematical calculations.

8. Write code for the following given scenario and add try-exception block to it. a. Program to divide two numbers;

= try:

numerator = 10, denominator = 0

result = numerator/denominator

print(result)

except:

print("Error: Denominator cannot be 0.")

c. Program to access an element in a list

= try:

if lst[3] == 'test':

pass # Do something

else:

pass # Do something

except IndexError:

pass # Handle the exception

d. Program to handle a specific exception ;

= Try and except statements are used to catch and handle exceptions in Python. Statements that can raise exceptions are kept inside the try clause and the statements that handle the exception are written inside except clause.

  we are trying to divide a number by 0. Here, this code generates an exception. To handle the exception, we have put the code,

try:

numerator = 10

denominator = 0

result = numerator/denominator

print(result)

except:

print("Error: Denominator cannot be 0.")

# Output: Error: Denominator cannot be 0.

e. Program to handle any exception;

= x = 5

y = "hello"

try:

    z = x + y

except TypeError:

    print("Error: cannot add an int and a str")

# output : "Error: cannot add an int and a str".