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

Ans1. Try exception block is used to handle run time errors so that application can run smoothly and if required we can show error message to the end users.

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

Ans2.

def my\_try\_except():

try:

print("My name is Ashish")

except:

print("There are some error occured")

my\_try\_except()

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

Ans3. It will generate system error message and show which exception type is required to handle.

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

Ans4. Bare except block will handle all kind of exception which we have not thought at the time of coding.

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

Ans5. Yes, we can have nested try-except blocks.

def my\_try\_except():

try:

1/0

try:

lst=[1,2,3,4,5]

lst[5]

except IndexError:

print('Index out of bound')

except ArithmeticError:

print("You have just made an Arithmetic error")

my\_try\_except()

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

Ans6. Yes, we can use multiple except block,

def my\_try\_except():

try:

1/0

except ArithmeticError:

print("You have just made an Arithmetic error")

except IndexError:

print('Index out of bound')

my\_try\_except()

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

a. EOFError

b. FloatingPointError

c. IndexError

d. MemoryError

e. OverflowError

f. TabError

g. ValueError

Ans7. Here are details:

1. EOFError : This error occurs when Python has reached the end of user input without receiving any input.
2. FloatingPointError : It is difficult to represent some decimal number in binary, so in many cases, it leads to small roundoff errors.
3. IndexError : This error occurs when an attempt is made to access an item in a list at an index which is out of bounds.
4. MemoryError : A MemoryError means that the interpreter has run out of memory to allocate to Python program.
5. OverflowError : This error occurs when any operations or variable storing value beyond its limit.
6. TabError : occurs when we mix tabs and spaces in the same code block
7. ValueError: occurs when a function receives an argument of the correct data 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

b. Program to convert a string to an integer

c. Program to access an element in a list

d. Program to handle a specific exception

e. Program to handle any exception

Ans8.

a. Program to divide two numbers

try:

10/12

except ArithmeticError as e:

print(e)

b. Program to convert a string to an integer

try:

int1=0

str1='1'

int1=int(str1)

print(type(int1))

print(type(str1))

except ArithmeticError as e:

print(e)

c. Program to access an element in a list

try:

lst=[1,2,3,4]

for i in lst:

print(i)

except:

print('Some error occured')

d. Program to handle a specific exception

try:

numerator = 10

denominator = 0

result = numerator/denominator

print(result)

except:

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

e. Program to handle any exception

try:

print('Code successfully run.')

except:

print('Some error occured')