Exception Handling

`Q-1`: You are given a function definition. There might be several issues on execution of this function. You are asked to do exception handling for diffrent errors that this function goes in to `without altering this function`. And print error text.

Function parameters `l -> list, s -> could be anything`

```
def function(l: list, s, **args):
    last_element = l[-1]

    l[int(s)]=10
    any_element = l[int(s)+10]
    l[s]=10

    res = sum(l)

    p = args['p']
    # print(p)
    return res/last_element * p + any_element
```

Check for different function calls:-

```
function([1,2,1], 12)
function([1,2,1]*9, '1-2')
function([1,'2',1]*9, 12)
function([1,'2',1]*9, 12)
function([1,2,0]*9, 12  )
function([1,2,1]*9, 12, p=None)
function([1,2,0]*9, 12, p=10)
```

```
# Write code here
def function(l: list, s, **args):
    last_element = l[-1]

    l[int(s)]=10
    any_element = l[int(s)+10]
    l[s]=10

    res = sum(l)

    p = args['p']
    # print(p)
    return res/last_element * p + any_element
try:
    res = function([1,2,1]*9, 12, p=10)
except IndexError as i:
    print(type(i))
    print(i)
except ValueError as v:
    print(type(v))
    print(v)
except TypeError as t:
    print(type(t))
    print(t)
except KeyError as k:
    print(type(k))
    print(k)
except ZeroDivisionError as z:
    print(type(z))
    print(z)
else:
    print("Result:", res)
finally:
    print("Thank you")
```

```
Result: 452.0
Thank you
```

Q-2: You are given a code snippet. There might be several issues on execution of this code. You are asked to do exception handling for diffrent errors, condition is what ever happens we need to execute last line printing correct result of `sum of elements`.

List have elemnts as any no of `key-pair dict with key as list index and value as any integer`, `integers` and `numeric-strings`. There is always only one element in the dict.

```
l = [{0:2},2,3,4,'5', {5:10}]
# For calculating sum of above list
s=0
for i in range(len(l)):
    #You can Edit code from here
    s += l[i].get(i)
    s += l[i]
    s += int(l[i])


print(s)
```

```
# Write code here
l = [{0:2},2,3,4,'5', {5:10}]
# For calculating sum of above list
s=0
for i in range(len(l)):
    try:
        s += l[i]
    except TypeError:
        try:
            s += l[i].get(i)
        except AttributeError:
            s += int(l[i])
print(s)
```

```
26
```

Q-3: : File Handling with Exception handling

Write a program that opens a text file and write data to it as "Hello, Good Morning!!!". Handle exceptions that can be generated during the I/O operations. Do not show the success message on the main exception handling block (write inside the else block).

```
# write code here
try:
    with open("text_file.txt", "w") as f:
        f.write("Hello, Good Morning!!!")
except IOError:
    print("Error working with file...")
else:
    print("File written successfully")
```

```
File written successfully
```

Q-4 : Number game program.

Write a number game program. Ask the user to enter a number. If the number is greater than number to be guessed, raise a **ValueTooLarge** exception. If the value is smaller the number to be guessed the, raise a **ValueTooSmall** exception and prompt the user to enter again. Quit the program only when the user enters the correct number. Also raise **GuessError** if user guess a number less than 1.

```
# Write code here
import random

class ValueTooLarge(Exception):
    def display(self):
        print("Input value is too large")

class ValueTooSmall(Exception):
    def display(self):
        print("Input value is too small")

class GuessError(Exception):
    def display(self):
        print("Guess the number between 1 and 100")

num = random.randint(1, 100)
while 1:
    try:
        guess = int(input("Enter a number:"))

        if guess < 1:
            raise GuessError

        if guess == num:
            print("Great You succeeded...")
            break
        if guess < num:
            raise ValueTooSmall
        if guess > num:
            raise ValueTooLarge
    except ValueTooSmall as s:
        s.display()
    except ValueTooLarge as l:
        l.display()
    except GuessError as g:
        g.display()
```

```
Enter a number: 5
Input value is too small
Enter a number: 1
Input value is too small
Enter a number: 99
Input value is too large
Enter a number: 55
Input value is too small
Enter a number: 65
Input value is too large
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_15804\983881989.py in <module>
     17 while 1:
     18     try:
---> 19         guess = int(input("Enter a number:"))
     20
     21         if guess < 1:

C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\kernelbase.py in raw_input(self,
prompt)
   1175                 "raw_input was called, but this frontend does not support input
requests."
   1176             )
-> 1177         return self._input_request(
   1178             str(prompt),
   1179             self._parent_ident["shell"],

C:\ProgramData\Anaconda3\lib\site-packages\ipykernel\kernelbase.py in
_input_request(self, prompt, ident, parent, password)
   1217             except KeyboardInterrupt:
   1218                 # re-raise KeyboardInterrupt, to truncate traceback
-> 1219                 raise KeyboardInterrupt("Interrupted by user") from None
   1220             except Exception:
   1221                 self.log.warning("Invalid Message:", exc_info=True)
```

Q-5: Cast vote

Write a program that validate name and age as entered by the user to determine whether the person can cast vote or not. To handle the age, create **InvalidAge** exception and for name, create **InvalidName** exception. The name will be invalid when the string will be empty or name has only one word.

Example 1:

Input:

```
Enter the name:              goransh singh
Enter the age: 25
```

Output:

```
 Goransh Singh  Congratulation !!! You can vote.
```

```
# Write code here
class InvalidAge(Exception):
    def display(self):
        print("Sorry!! Age cannot be below 18. Please come after that")
class InvalidName(Exception):
    def display(self):
        print("Please enter a valid name...")
try:
    name = input("Enter your name: ")
    if len(name) == 0 or len(name.split()) < 2:
        raise InvalidName

    age = int(input("Enter your age: "))
    if age < 18:
        raise InvalidAge
except InvalidName as i:
    i.display()
except InvalidAge as a:
    a.display()
else:
    print("Your vote is accepted successfully")
finally:
    print("Thank You")
```

```
 Enter your name:  Raj sdfsdf
 Enter your age:  19
 Your vote is accepted successfully
 Thank You
```

Q-6 : Write a python function which infinitely prints natural numbers in a single line. Raise the **StopIteration** exception after displaying first 20 numnbers to exit from the program.

```
# Write code here
def display(n):
    i=0
    while True:
        try:
            n+=1
            i+=1
            if i==21:
                raise StopIteration
        except StopIteration:
            break
        else:
            print(n, end =' ')
display(100)
```

```
 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
```

Start coding or generate with AI.