Exceptions Handling

```
# ===================================
# == Errors and Exceptions Raising ==
# ===================================
# [1] Exception: is a runtime error reporting mechanism
# [2] Exception gives you the message to understand the problem
# [3] Traceback gives you the line to look for the code in this line
# [4] Exceptions have types (SyntaxError, IndexError, KeyError, Etc, ...)
# [5] Exceptions list https://docs.python.org/3/library/exceptions.html
# [6] "raise" keywork used to raise your own exceptions

# ===============
# == Note That ==
# ===============
# - the code will not complete unless the exception solved

x = -10
if x < 0:
    # any command after exception will not print
    raise Exception(f"The Number {x} Is Less Than Zero")
else:
    print(f"The Number {x} Is Bigger Than Zero")
print("Print Message After If Condition")


y = 'osama'
if type(y) != int:
    raise ValueError("Only Numbers Allowed")
else:
    print(y)
```

```
# ================================
# == Try, Except, Else, Finally ==
# ================================
# Try: Test The Code For Error
# Except: Handle The Error
# Else: If No Error
# Finally: Run The Code, run whatever happens

try:    # try the code and test the error
    number = int(input("Write Number Here: "))
    print(f"result from try: {number*10}")

except: # handle the error if found
    print("This Is Not Integer, Only Integer Allowedd!")

else:   # if there is no error
    print(f"result from else: number*2")

# ===========================================================
```

```python
# can write the code without "else"
# usually write the code of "else" in try
try:
    num = int(input("Enter Your Number: "))
    print(f"The Result Is: {num*10}")
except:
    print("Only Integer Allowed")

# ===========================================================

# "except" will print to all error occurs on the "try"
try:
    print(10 / 0)       #
except:
    print("can not divide")

# ===========================================================

# can determine the type of the exception
try:
    print(int("Hello"))     # Value Error
    print(name)             # Identifier Not Found
    print(10 / 0)           # Can Not Divide By Zero

except ZeroDivisionError:
    print("Can Not Divide By Zero")

except NameError:
    print("Identifier Not Found")

except ValueError:
    print("Value Error")

# for global error that not determined on the code
except:
    print("Error Happens")
```