# Day 14 – Try & Catch

# • When an error occurs, or exception as we call it, Python will normally stop and generate an error message. These exceptions can be handled using the try statement:

# try:

# print(a)

# except:

# print("An exception occurred")

# . If you want to execute a special block of code for a special kind of error like the if else block we can specify the error.

# try:

# print(player)

# except NameError:

# print("player not defined")

# except:

# print("End of program")

# use the else keyword to define a block of code to be executed if no errors were raised

# try:

# print(x)

# except:

# print(“Exception”)

# else:

# print(“End of code”)

# finally block, if specified, will be executed regardless if the try block raises an error or not

# print(x)

# except:

# print("Something went wrong")

# finally:

# print("The 'try except' is finished")

# Raise an exception

# As a Python developer you can choose to throw an exception if a condition occurs.

# To throw (or raise) an exception, use the raise keyword.

# x = -1

# if x < 0:

# raise Exception("Sorry, no numbers below zero")

# Exercise:

# List down all the error types and check all the errors using a python program for all errors

# Design a simple calculator app with try and except for all use cases

# print one message if the try block raises a NameError and another for other errors

* + - When try-except scenario is not required?
    - Try getting an input inside the try catch block