# Coding and Solutioning

## Exception Handling

| | |
|---|---|
| Date | 19 November 2022 |
| Team ID | PNT2022TMID51098 |
| Project Name | Real-Time River Water Quality Monitoring and Control System |
| Maximum Marks | 2 Marks |

### Exception Handling

- Many built-in exceptions in Python are thrown when a problem is encountered by your programme (something in the programme goes wrong).
- The Python interpreter stops the running process and passes control to the calling process until these exceptions are addressed. The software will crash if it is not addressed.

Let's take a software where function A calls function B, which then calls function C as an example. An exception that occurs in function C but isn't handled there is passed to B before being handled by A.

If the mistake is never addressed, a notification is shown and our software abruptly and unexpectedly stops.

### Catching Exceptions

A try statement in Python can be used to manage exceptions. The try clause contains the crucial operation that can cause an exception. The except clause contains the code that manages exceptions. Thus, once we have identified the exception, we may decide what actions to take.

Here is an easy illustration.

```
try:
        deviceOptions = {"org": organization,
            "type": deviceType, "id": deviceId,
            "auth-method": authMethod,"auth-token":
            authToken}
        deviceCli = ibmiotf.device.Client
            (deviceOptions)
except Exception as e:
    print("Caught exception connecting device: %s"
        % str(e))
    sys.exit()
    deviceCli.connect()
```