CERN-HSF GSoC 2019

Exercise for Candidate Students for the following projects:

- **Monitoring DIRAC components [link]**

The projects are included in the Google Summer of Code (GSoC) program 2019 and offered by the EP-LBC group at CERN (LHCb computing group).

This exercise is divided into 3 tasks, which have to be done sequentially. There is also a fourth task listed here, which is bonus task. In order to complete all of them, the student will need to know and combine a set of technologies that are important for the project, namely Python and ElasticSearch.

Please follow the guidelines below to go through the exercise and work at a pace that suits you. Start with the first task and, if you finish it and have some more time, you go for the second one. If you finish the second one, go for the third. If you still have time and the will, do also the fourth . Do not worry if you cannot complete all of them, and do not hesitate to ask us, the mentors, any question you might have.

Task 1

This task is about creating a unit test, using python 2.7 and pytest. In order to complete the task, please follow these steps:

- use Linux or MacOS
- install python 2.7 if not already installed. The last version is 2.7.15, but any 2.7.x version would be enough, but please prefer 2.7.9+
- install virtualenv if not already installed, or alternatively work with miniconda
- create, with virtualenv, an isolated python environment where you would install, using pip:
    - pytest
- code a python module that contains a unit test based on pytest that:
    - creates a file on disk
    - verifies (assert) that the file has been created
    - deletes the file
    - verifies (assert) that the file is not anymore on disk
- run the unit test you coded above using pytest

Deliverable:

- The python code of the unit test
- The most important commands you ran for running it

Task 2

- install *mock,* using pip, in the virtualenv (or conda if you did you conda instead of virtualenv)
- Consider the following unit test:

```
""" Simple test on using mock
"""

import mock
import pytest


#########################
# function to test

def myFunction(objectIn):
  """ what you are supposed to test
  """
  return objectIn.aMethodToMock() + 2

#########################
# Actual test


@pytest.fixture
def objectMock():
  """ A fixture to create a fake object as we want it"""
  fakeObject = mock.MagicMock()
  return fakeObject


def test_myFunction(objectMock):
  """ Test myFunction """
  assert myFunction(objectMock) == 5
```

- This test will fail. Why?
- Can you fix this test? Just add 1 line of code, use *mock*.

Deliverable: the python code of the unit test above, completed with the missing line

Task 3

- install *psutil* inside the virtualenv/conda
- create a simple python module that:
  - gets the current process ID
  - discovers the name, the status, and the number of threads run by the process
  - dumps these info in a *json* file

Deliverable:

- The python code of the module

Task 4 [BONUS]

- install ElasticSearch 6 and run it
- install in your virtual python environment the python ElasticSearch client: https://github.com/elastic/elasticsearch-py
- Using the pytest framework, create an integration test that:
    - connects to the running instance
    - creates an Index in elasticSearch
    - verifies that the index has been created
    - lists, then deletes all the indexes

    - verifies that the index has been deleted
    - re-creates the index, then add the content of the JSON file created in Task 3
    - retrieves it, verifies its content

Deliverable:

- The python code of the test