Quiz 2 - Computational Physics II

17/20

NAME: Alan Palma Travez _____ **SCORE:**

Date: Thursday 20 March 2025 **Duration:** 45 minutes

Credits: 20 points (4 questions) **Type of evaluation:** LAB

Provide short and concise answers to the following items. Code syntax should be clear.

1. **(5 points) Python classes**
   (a) Provide a simple code snippet of a Python class showing how it is defined and its components.
   (b) Explain the difference between an instance attribute and a class attribute in a Python class.

-0.5

a)
```
class ClassExample():                          # class name
    """
    This is a simple class
    """
    course = "CPII"                            # class attribute; points to the memory
    def __init__(self, name):                  # init function initializes some parameters that can be here. It runs when the class is instanced.
        self.name = name                       # instance attribute
    def show_name(self):                       # this is a method which is within the class for a defined task and can be called with the instance of the class or within the class by other methods.
        print(self.name)
```

Usage:
```
>> a = ClassExample("Alan")
>> a.showname()
>> "Alan"
```

An instance attribute is an variable/function that is defined when the class is instanced in an object, a = ClassExample("Alan"), while a class attribute is a variable/function defined within the class.?

Both are defined within a class

cls

2. **(5 points) Python decorators**
   (a) What is a decorator in Python, and what is its purpose?
   (b) Provide a simple code example of a decorator and explain what it does.

-0.5

a) A decorator in Python is a function that accepts another function as an argument. Decorators are used for attributing new responsabilities or functionalities to a method. The user can create his/her own decorators or use the built-in decorators.

b)
```
def mydecorator(fun):                          # Docstring
    fun()                                      # extra function. args are missing
    print("Quiz number two")
                                               # return statement?
```

→ This decorator adds the message "Quiz number two" to the functionalities of the argument fun.

```
@mydecorator
def func_ex():
    print("This is the ", 1+1)
```

The resulting message after calling this function will be "This is the Quiz number two".

3. **(5 points) Python packages**

(a) Describe the typical directory structure of a well-designed Python package.

(b) Describe the primary purpose of the `argparse` module.

-0.5

a) The standarized structure is:

mymodule
- mymodule ← This is actually the module
  - __init__.py ← init script to python recognize the module and initialize.
  - main.py ← main script with all the functions/classes
- README.md
- test.py → testing scrip to detect errors and bugs.
- setup.py → script used to install the module.

b) argpase module is used to create line-style comand for running a script or a module. This tool creates automatically a help option where the usage is showed to the user. ok, but it's also used to pass arguments to the module from the CLI, via flags.

4. **(5 points) Testing Python modules**

(a) Why is it important to add testing classes to Python modules?

(b) Write set-up and tear-down test classes using `pytest` for the class below. Your test class should check if the `time_of_flight` method returns an expected value.

-1.5

```python
import numpy as np
class ParabolicMotion:
    """ A class to calculate the flight time of a projectile."""
    def __init__(self, v0, angle):
        """Initial velocity (v0 in m/s) and launch angle (degrees)."""
        self.v0 = v0
        self.angle = np.radians(angle)
        self.g = 9.81  # Gravity (m/s^2)
    def time_of_flight(self):
        """Returns the total time the projectile stays in the air."""
        return (2 * self.v0 * np.sin(self.angle)) / self.g
```

a) It is important because it helps the developer to identify mistakes and bugs in the code with anticipation. It is also used for running in the background when an actualization is made so it should be ligth. This also is used to avoid mistakes when the user don't provide the rigth parameters.

b)
```
import pytest
class TestClass():
    @classmethod decorator
    def tear_down_test(self):                    ← setup_method/class is missing.
        setup?
        self.v0 = 9.81 ?   ✗ del objects
        self.ang = 90.0 → free fall ✓.
    def test1_parabolic(self):
        pm = ParabolicMotion(self.v0, self.angle)
        time = pm.time_of_flight
        if not time == 2.0:        ← this value is incorrect for validation.
            raise TypeError("Time is not computed correctly").
    @ decorator is missing
    def tear_down_test():
        print("Test executed").
```

The if statement should be replaced with the correct(s) pytest methods. but I don't remember. the functions that we use in class. ok.