

## Finals Task 2. Inheritance

### Problem School Performance

Note: You are to create 4 separate python files for this task:

- performer.py(base class)
- singer.py(sub class)
- dancer.py(sub class)
- test\_class.py – following the required test cases

In a school musical performance, different types of performers participate. For this program, we will be implementing the performers.

#### Base Class - Performer:

- Properties:
  - `name` (type: str): Represents the name of the performer.
  - `age` (type: int): Represents the age of the performer.
- Constructor:
  - `__init__(self, name: str, age: int)`: Initializes the `name` and `age` properties.
- Getters
  - `get_name(self) -> str`: Returns the name
  - `get_age(self) -> int`: Returns the age

#### Subclass - Singer:

- Inherits From: `Performer`
- Additional Property:
  - `vocal_range` (type: str): Represents the vocal range of the singer.
- Constructor:
  - `__init__(self, name: str, age: int, vocal_range: str)`: Initializes the `name` and `age` properties by calling the parent class's constructor and sets the `vocal_range` property.
- Getter:
  - `get_vocal_range(self) -> str`: Returns the vocal range of the singer.
- Method:
  - `sing(self) -> None`: Prints "{name} is singing with a {vocal\_range} range."

### Subclass - Dancer:

- Inherits From: **Performer**
- Additional Property:
  - **dance\_style** (type: str): Represents the dance style of the dancer.
- Constructor:
  - **\_\_init\_\_(self, name: str, age: int, dance\_style: str)**: Initializes the **name** and **age** properties by calling the parent class's constructor and sets the **dance\_style** property.
- Getter:
  - **get\_dance\_style(self) -> str**: Returns the dance style of the dancer.
- Method:
  - **dance(self) -> None**: Prints "{name} is performing {dance\_style} dance."

### Sample output for the Test Class

#### Test Cases

##### Test case 1

Should return [ 'John', 25 ] when invoking the methods [ get\_name(), get\_age() ] of the Performer class with properties { Name: 'John' , Age: 25 }.

##### Test case 2

Should return [ 'Emily', 28, 'Ballet' ] when invoking the methods [ get\_name(), get\_age(), get\_dance\_style() ] of the Dancer class with properties { Name: 'Emily' , Age: 28, Dance Style: 'Ballet' }.

##### Test case 3

Should return 'Emily is performing Ballet dance.' when invoking the dance() method of the Dancer class with properties { Name: 'Emily' , Age: 28, Dance Style: 'Ballet' }.

##### Test case 4

Should make Dancer class a subclass of Performer class.

##### Test case 5

Should return [ 'Linda', 35, 'Soprano' ] when invoking the methods [ get\_name(), get\_age(), get\_vocal\_range() ] of the Singer class with properties { Name: 'Linda' , Age: 35, Vocal Range: 'Soprano' }.

##### Test case 6

Should return 'Linda is singing with a Soprano range.' when invoking the sing() method of the Singer class with properties { Name: 'Linda' , Age: 35, Vocal Range: 'Soprano' }.